Package 'TreeSearch'

June 3, 2019

Title Phylogenetic Tree Search Using Custom Optimality Criteria

Version 0.3.2

Date 2019-06-03

License GPL (>= 3)

Copyright Incorporates C/C++ code from: ape by Emmanuel Paradis;
phangorn by Klaus Scheip <doi:10.1093/bioinformatics/btq706>;
Morphy Phylogenetic Library by Martin Brazeau
<https://github.com/mbrazeau/MorphyLib> (GPL3)

Description Searches for phylogenetic trees that are optimal using a user-defined criterion.
Implements Profile Parsimony (Faith and Trueman, 2001) <doi:10.1080/10635150118627>, and
Successive Approximations (Farris, 1969) <doi:10.2307/2412182>.
Handles inapplicable data using the algorithm of Brazeau, Guillerme and Smith
(2019) <doi:10.1093/sysbio/syy083>.

URL https://github.com/ms609/TreeSearch

BugReports https://github.com/ms609/TreeSearch/issues

Depends R (>= 3.4.0)

Imports ape (>= 5.0), clue, colorspace, memoise, phangorn (>= 2.2.1),
R.cache, Rdpack, stats

Suggests knitr, Rcpp, rmarkdown, shiny, testthat, xlsx

RdMacros Rdpack

LazyData true

ByteCompile true

LinkingTo Rcpp

Encoding UTF-8

Language en-GB

VignetteBuilder knitr

RoxygenNote 6.1.1

NeedsCompilation yes

Author Martin R. Smith [aut, cre, cph]
(<https://orcid.org/0000-0001-5660-1727>),
Emmanuel Paradis [cph] (<https://orcid.org/0000-0003-3092-2199>),
Klaus Schliep [cph] (<https://orcid.org/0000-0003-2941-0161>),
Martin Brazeau [cph] (<https://orcid.org/0000-0002-0650-1282>)

Maintainer Martin R. Smith <martin.smith@durham.ac.uk>

Repository CRAN

Date/Publication 2019-06-03 15:30:03 UTC

R topics documented:

AddTip *Add a tip to a phylogenetic tree*

Description

AddTip adds a tip to a phylogenetic tree at a specified location.

Usage

```
AddTip(tree, where, label)
```

Arguments

tree	A tree of class phylo.
where	The node or tip that should form the sister taxon to the new node. To add a new tip at the root, use "where = 0";
label	A character string providing the label the new tip.

Details

AddTip extends bind.tree, which cannot handle single-taxon trees.

Value

This function returns a tree of class phylo with an additional tip at the desired location.

Author(s)

Martin R. Smith

See Also

bind.tree

nodelabels

Examples

```
{
  library('ape')
  plot(tree <- rtree(10, br=NULL)); nodelabels(); nodelabels(15, 15, bg='green'); dev.new()
  plot(AddTip(tree, 15, 'NEW_TIP'))
}
```

`AllAncestors` *List all ancestral nodes*

Description

`AllAncestors` lists ancestors of each parent node in a tree

Usage

```
AllAncestors(parent, child)
```

Arguments

`parent`	the first column of the edge matrix of a tree of class `phylo`, i.e. `tree$edge[, 1]`
`child`	the second column of the edge matrix of a tree of class `phylo`, i.e. `tree$edge[, 2]`

Details

Note that the tree's edges must be listed in an order whereby each entry in `tr$edge[, 1]` (with the exception of the root) has appeared already in `tr$edge[, 2]`.

Value

This function returns a list. Entry i contains a vector containing, in order, the nodes encountered when traversing the tree from node i to the root node. The last entry of each member of the list will therefore be the root node, with the exception of the entry for the root node itself, which will be NULL.

Author(s)

Martin R. Smith

Examples

```
tr <- ape::rtree(20, br=NULL)
edge <- tr$edge
AllAncestors(edge[, 1], edge[, 2])
```

`AllSplitPairings` *All split pairings*

Description

Calculates the variation of (clustering) information (Meila 2007) for each possible pairing of non-trivial splits, and tabulates the number of pairings with each similarity.

Usage

```
AllSplitPairings(n)

SplitPairingInformationIndex(n)
```

Arguments

`n`	Integer specifying the number of terminal taxa.

Details

Trivial splits – those that divide one or zero taxa from the rest – are not considered.

Value

`AllSplitPairings` returns a named vector, specifying the number of split pairings producing the variation of information given (in bits) in the name. Splits AB:CD and CD:AB are treated as distinct, so division of all values by four is justified in cases where unique pairings only are required.

`SplitPairingInformationIndex` returns a table listing the possible values of the variation of information for splits with `n` terminals, and the clustering information (*sensu* Smith forthcoming) associated with a pairing that has the given variation of information.

Functions

- `SplitPairingInformationIndex`: Lookup table listing split pairing information

Author(s)

Martin R. Smith

References

- MeilÄ*f* M (2007). "Comparing clusterings—an information based distance." *Journal of Multivariate Analysis*, **98**(5), 873–895. doi: 10.1016/j.jmva.2006.11.013.
- Smith MR (2019). "Information theoretic Generalized Robinson-Foulds metrics for measuring the distance between phylogenetic trees." *Forthcoming*.

Examples

```
AllSplitPairings(5)
SplitPairingInformationIndex(5)
```

`AllSPR` *All SPR trees*

Description

All SPR trees

Usage

```
AllSPR(parent, child, nEdge, notDuplicateRoot, edgeToBreak)
```

Arguments

`parent`	the first column of the edge matrix of a tree of class `phylo`, i.e. `tree$edge[, 1]`
`child`	the second column of the edge matrix of a tree of class `phylo`, i.e. `tree$edge[, 2]`
`nEdge`	integer specifying the number of edges of a tree of class `phylo`, i.e. `dim(tree$edge)[1]`
`notDuplicateRoot`	logical vector of length nEdge, specifying for each edge whether it is the second edge leading to the root (in which case its breaking will be equivalent to breaking the other root edge... except insofar as it moves the position of the root.)
`edgeToBreak`	(optional) integer specifying the index of an edge to bisect/prune, generated randomly if not specified. Alternatively, set to `-1` to return a complete list of all trees one step from the input tree.

Value

a list of edge matrices for all trees one SPR rearrangement from the starting tree

Author(s)

Martin R. Smith

ApeTime *Ape Time*

Description

Reads the time that an ape tree was modified from the comment in the nexus file

Usage

```
ApeTime(filename, format = "double")
```

Arguments

filename	Character string specifying path to the file
format	Format in which to return the time: 'double' as a sortable numeric; any other value to return a string in the format YYYY-MM-DD hh:mm:ss

Value

The time that the specified file was created by ape.

Author(s)

Martin R. Smith

AsBinary *Convert a number to binary*

Description

Provides a (reversed) binary representation of a decimal integer

Usage

```
AsBinary(x)
```

Arguments

x	Decimal integer to be converted to binary bits

Details

Provides an array corresponding to binary digits 1, 2, 4, 8, 16, ...

Binary number 0100 (= decimal 4) will be represented as 0 0 1.

Value

An array corresponding to binary digits 1, 2, 4, 8, 16, ...

'Leading zeros' are not included.

Author(s)

Martin R. Smith, adapted from code posted to R mailing list by Spencer Graves

Examples

```
AsBinary(4)  # 0 0 1
AsBinary(10) # 0 1 0 1
```

---

brewer *Brewer palettes*

Description

A list of eleven Brewer palettes containing one to eleven colours that are readily distinguished by colourblind viewers, followed by a twelfth 12-colour palette adapted for colour blindness.

Usage

```
brewer
```

Format

An object of class `list` of length 12.

Source

ColourBrewer2.org Martin Krzywinski

`CollapseNode` *Collapse nodes on a phylogenetic tree*

Description

Collapses specified nodes or edges on a phylogenetic tree, resulting in polytomies.

Usage

```
CollapseNode(tree, nodes)

CollapseEdge(tree, edges)
```

Arguments

`tree`	A tree of class `phylo`.
`nodes, edges`	Integer vector specifying the nodes or edges in the tree to be dropped. (Use `nodelabels` or `edgelabels` to view numbers on a plotted tree.)

Value

`tree`, with the specified nodes or edges collapsed. The length of each dropped edge will (naively) be added to each descendant edge.

Author(s)

Martin R. Smith

Examples

```
library(ape)
set.seed(1)

tree <- rtree(7)
par(mfrow=c(2, 1), mar=rep(0.5, 4))
plot(tree)
nodelabels()
edgelabels(round(tree$edge.length, 2), cex=0.6, frame='n', adj=c(1, -1))

newTree <- CollapseNode(tree, c(12, 13))
plot(newTree)
nodelabels()
edgelabels(round(newTree$edge.length, 2), cex=0.6, frame='n', adj=c(1, -1))
```

congreveLamsdellMatrices *100 simulated data matrices*

Description

Contains the 100 simulated matrices generated by Congreve & Lamsdell (2016) using a heterogeneous Markov-k model, generated from the referenceTree topology, with all branches sharing an equal length.

Usage

```
congreveLamsdellMatrices
```

Format

A list with 100 entries, each comprising a phyDat object of 55 characters for 22 taxa

Source

https://datadryad.org/resource/doi:10.5061/dryad.7dq0j

References

Congreve CR, Lamsdell JC (2016). "Implied weighting and its utility in palaeontological datasets: a study using modelled phylogenetic matrices." *Palaeontology*, **59**, 447-465. doi: 10.1111/pala.12236.
Congreve CR, Lamsdell JC (2016). "Data from: Implied weighting and its utility in palaeontological datasets: a study using modelled phylogenetic matrices." *Dryad Digital Repository*, doi:10.5061/dryad.7dq0j. doi: 10.5061/dryad.7dq0j.

Examples

```
  data('referenceTree')
  data('congreveLamsdellMatrices')
  ## Not run: ProfileScore(referenceTree,
   PrepareDataProfile(congreveLamsdellMatrices[[17]]))
## End(Not run)
```

`ConsensusWithout` *Consensus without taxa*

Description

Displays a consensus plot with selected taxa excluded.

Usage

```
ConsensusWithout(trees, tip, ...)

MarkMissing(tip, position = "bottomleft", ...)
```

Arguments

`trees`	A list of phylogenetic trees, of class `multiPhylo` or `list`
`tip`	A character vector specifying the names (or numbers) of tips to drop (using ape::drop.tip)
`...`	Additional parameters to pass on to ape::consensus or legend
`position`	Where to plot the missing taxa. See legend for options.

Details

A useful way to gain resolution if a few wildcard taxa obscure a consistent set of relationship.

Value

A consensus tree without the excluded taxa

Functions

- `MarkMissing`: Adds missing taxa to a plotted consensus tree

Author(s)

Martin R. Smith

Martin R. Smith

DescendantEdges *Descendant Edges*

Description

Quickly identifies edges that are 'descended' from a particular edge in a tree

Usage

```
DescendantEdges(edge, parent, child, nEdge = length(parent))

AllDescendantEdges(parent, child, nEdge = length(parent))
```

Arguments

edge	number of the edge whose child edges are required
parent	the first column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 1]
child	the second column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 2]
nEdge	number of edges (calculated from length(parent) if not supplied)

Value

DescendantEdges returns a logical vector stating whether each edge in turn is a descendant of the specified edge (or the edge itself)

AllDescendantEdges returns a matrix of class logical, with row N specifying whether each edge is a descendant of edge N (or the edge itself)

Functions

- AllDescendantEdges: Quickly identifies edges that are 'descended' from each edge in a tree

DoubleFactorial *Double Factorial*

Description

Double Factorial

Usage

```
DoubleFactorial(n)

LogDoubleFactorial(n)

LogDoubleFactorial.int(n)
```

Arguments

n	Vector of integers.

Value

Returns the double factorial, n x (n - 2) x (n - 4) x (n - 6) x ...

Functions

- `LogDoubleFactorial`: Returns the logarithm of the double factorial.
- `LogDoubleFactorial.int`: Slightly faster, when x is known to be length one and below 50001

Author(s)

Martin R. Smith

Examples

```
{
DoubleFactorial (-4:0) # Return 1 if n < 2
DoubleFactorial (2) # 2
DoubleFactorial (5) # 1 x 3 x 5
exp(LogDoubleFactorial.int (8)) # 2 x 4 x 6 x 8

}
```

`doubleFactorials` *Double factorials*

Description

A vector with pre-calculated values of double factorials up to 300!!, and the logarithms of double factorials up to 50 000!!.

Usage

```
doubleFactorials
```

Format

An object of class `numeric` of length 300.

Details

301!! is too large to store as an integer; use `logDoubleFactorials` instead.

DropSingleSplits *Drop Single Splits*

Description

Removes splits that pertain only to a single taxon from a splits object.

Usage

```
DropSingleSplits(split)
```

Arguments

split	A matrix in which each column corresponds to a bipartition split

Details

Bipartition splits are divisions, implied by each edge or node of an unrooted tree topology, that divide the taxa into two groups (one of which is a clade).

By default, a list of splits will include those that separate a single taxon (a leaf) from all others. Such splits are, by definition, present in all trees that contain that taxon; they are not of interest when comparing trees. This function removes such splits from a list of bipartitions.

Value

The input matrix, with any columns that separate only a single pendant tip removed.

Author(s)

Martin R. Smith

EdgeAncestry *EdgeAncestry*

Description

Descendant Edges

Usage

```
EdgeAncestry(edge, parent, child, stopAt = (parent == min(parent)))
```

Arguments

edge	number of the edge whose child edges are required
parent	the first column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 1]
child	the second column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 2]
stopAt	number of the edge at which the search should terminate; defaults to the root edges

Details

Quickly identifies edges that are 'ancestral' to a particular edge in a tree

Value

a logical vector stating whether each edge in turn is a descendant of the specified edge

Author(s)

Martin R. Smith

EnforceOutgroup *Force taxa to form an outgroup*

Description

Given a tree or a list of taxa, rearrange the ingroup and outgroup taxa such that the two are sister taxa across the root, without altering the relationships within the ingroup or within the outgroup.

Usage

```
EnforceOutgroup(tree, outgroup)
```

Arguments

tree	either a tree of class phylo, or a character vector listing the names of all the taxa in the tree, from which a random tree will be generated.
outgroup	a vector containing the names of taxa to include in the outgroup

Value

a tree where all outgroup taxa are sister to all remaining taxa, otherwise retaining the topology of the ingroup.

Author(s)

Martin R. Smith

Entropy *Entropy in bits*

Description

Reports the entropy of a vector of probabilities, in bits. Probabilities should sum to one. Probabilities equalling zero will be ignored.

Usage

```
Entropy(p)
```

Arguments

p	Numeric vector specifying probabilities of outcomes.

Value

Entropy of the specified probabilities, in bits

Author(s)

Martin R. Smith

Examples

```
Entropy(rep(0.5, 2)) # = 1
Entropy(c(1/4, 1/4, 0, 1/4, 1/4)) # = 2
```

Evaluate *Evaluate tree*

Description

Evaluate tree

Usage

```
Evaluate(tree, dataset, warn = TRUE)
```

Arguments

tree	A tree of class phylo.
dataset	A phylogenetic data matrix of class phyDat, whose names correspond to the labels of any accompanying tree.
warn	Boolean (TRUE/FALSE): display warnings when concavity functions are generated by approximation.

`Fitch` *Calculate parsimony score with inapplicable data*

Description

Uses code modified from the Morphy library to calculate a parsimony score in datasets that contain inapplicable data

Usage

```
Fitch(tree, dataset)
```

Arguments

`tree`	A tree of class `phylo`.
`dataset`	A phylogenetic data matrix of class `phyDat`, whose names correspond to the labels of any accompanying tree.

Value

This function returns the elements from a list containing:

- The total parsimony score
- The parsimony score associated with each character
- A matrix comprising character reconstructions for each node after the final pass

The elements to return are specified by the parameter `detail`. If a single element is requested (default) then just that element will be returned If multiple elements are requested then these will be returned in a list.

Author(s)

Martin R. Smith (using Morphy C library, by Martin Brazeau)

See Also

`TreeSearch`

Examples

```
data('inapplicable.datasets')
tree <- RandomTree(inapplicable.phyData[[1]])
result <- Fitch(tree, inapplicable.phyData[[1]])
```

FitchSteps *Fitch score*

Description

Fitch score

Usage

```
FitchSteps(tree, dataset)
```

Arguments

tree	A tree of class phylo.
dataset	A phylogenetic data matrix of class phyDat, whose names correspond to the labels of any accompanying tree.

Value

A vector listing the number of 'parsimony steps' calculated by the Fitch algorithm for each character. Inapplicable tokens are treated as per Brazeau, Guillerme and Smith (2017)

References

Brazeau MD, Guillerme T, Smith MR (2019). "An algorithm for morphological phylogenetic analysis with inapplicable data." *Systematic Biology*, in press. doi: 10.1093/sysbio/syy083.

Examples

```
{
}
```

ICSteps *Information Content Steps*

Description

This function estimates the information content of a character char when e extra steps are present, for all possible values of e.

Usage

```
ICSteps(char, ambiguousToken = 0, expectedMinima = 25,
  maxIter = 10000, warn = TRUE)
```

Arguments

`char`	The character in question.
`ambiguousToken`	Which token, if any, corresponds to the ambiguous token (?) (not yet fully implemented).
`expectedMinima`	sample enough trees that the rarest step counts is expected to be seen at least this many times.
`maxIter`	Maximum iterations to conduct.
`warn`	Boolean (`TRUE`/`FALSE`): display warnings when concavity functions are generated by approximation.

Details

Calculates the number of trees consistent with the character having *e* extra steps, where *e* ranges from its minimum possible value (i.e. number of different tokens minus one) to its maximum. The number of trees with no extra steps can be calculated exactly; the number of trees with more additional steps must be approximated. The function samples `n.iter` trees, or enough trees that the trees with the minimum number of steps will be recovered at least `expected.minima` times, in order to obtain precise results.

Author(s)

Martin R. Smith

References

Faith DP, Trueman JWH (2001). "Towards an inclusive philosophy for phylogenetic inference." *Systematic Biology*, **50**(3), 331–350. doi: 10.1080/106351501186627.

Examples

```
{
  # A character that is present in ten taxa and absent in five
  character <- c(rep(1, 10), rep(2, 5))
  ICSteps (character)
}
```

---

inapplicable.citations

Thirty Datasets with Inapplicable data

Description

These are the datasets used to evaluate the behaviour of the inapplicable algorithm in Brazeau, Guillerme and Smith (2018).

Usage

```
inapplicable.citations
```

Format

An object of class `character` of length 30.

Details

The name of each item corresponds to the datasets listed below. The value gives its citation.

Source

Agnarsson2004 AGNARSSON, I. 2004. Morphological phylogeny of cobweb spiders and their relatives (Araneae, Araneoidea, Theridiidae). Zoological Journal of the Linnean Society, 141, 447–626.

Capa2011 CAPA, M., HUTCHINGS, P., AGUADO, M. T. and BOTT, N. J. 2011. Phylogeny of Sabellidae (Annelida) and relationships with other taxa inferred from morphology and multiple genes. Cladistics, 27, 449–469.

DeAssis2011 DE ASSIS, J. E. and CHRISTOFFERSEN, M. L. 2011. Phylogenetic relationships within Maldanidae (Capitellida, Annelida), based on morphological characters. Systematics and Biodiversity, 9, 233–245.

OLeary1999 O'LEARY, M. A. and GEISLER, J. H. 1999. The position of Cetacea within Mammalia: phylogenetic analysis of morphological data from extinct and extant taxa. Systematic Biology, 48, 455–490.

Rousset2004 ROUSSET, V., ROUSE, G. W., SIDDALL, M. E., TILLIER, A. and PLEIJEL, F. 2004. The phylogenetic position of Siboglinidae (Annelida) inferred from 18S rRNA, 28S rRNA and morphological data. Cladistics, 20, 518–533.

Sano2011 SANO, M. and AKIMOTO, S.-I. 2011. Morphological phylogeny of gall-forming aphids of the tribe Eriosomatini (Aphididae: Eriosomatinae). Systematic Entomology, 36, 607–627.

Sansom2010 SANSOM, R. S., FREEDMAN, K., GABBOTT, S. E., ALDRIDGE, R. J. and PURNELL, M. A. 2010. Taphonomy and affinity of an enigmatic Silurian vertebrate, *Jamoytius kerwoodi* White. Palaeontology, 53, 1393–1409.

Schulze2007 SCHULZE, A., CUTLER, E. B. and GIRIBET, G. 2007. Phylogeny of sipunculan worms: A combined analysis of four gene regions and morphology. Molecular Phylogenetics and Evolution, 42, 171–92.

Shultz2007 SHULTZ, J. W. 2007. A phylogenetic analysis of the arachnid orders based on morphological characters. Zoological Journal of the Linnean Society, 150, 221–265.

Wetterer2000 WETTERER, A. L., ROCKKMAN, M. V. and SIMMONS, N. B. 2000. Phylogeny of phyllostomid bats (Mammalia: Chiroptera): data from diverse morphological systems, sex chromosomes, and restriction sites. Bulletin of the American Museum of Natural History, 248, 1–200.

Wills2012 WILLS, M. A., GERBER, S., RUTA, M. and HUGHES, M. 2012. The disparity of priapulid, archaeopriapulid and palaeoscolecid worms in the light of new data. Journal of Evolutionary Biology, 25, 2056–2076.

Aguado2009 AGUADO, M. T. and SAN MARTIN, G. 2009. Phylogeny of Syllidae (Polychaeta) based on morphological data. Zoologica Scripta, 38, 379–402.

Aria2015 ARIA, C., CARON, J. B. and GAINES, R. 2015. A large new leanchoiliid from the Burgess Shale and the influence of inapplicable states on stem arthropod phylogeny. Palaeontology, 58, 629–660.

Asher2005 ASHER, R. J. and HOFREITER, M. 2006. Tenrec phylogeny and the noninvasive extraction of nuclear DNA. Systematic biology, 55, 181–94.

Baker2009 BAKER, W. J., SAVOLAINEN, V., ASMUSSEN-LANGE, C. B., CHASE, M. W., DRANSFIELD, J., FOREST, F., HARLEY, M. M., UHL, N. W. and WILKINSON, M. 2009. Complete generic-level phylogenetic analyses of palms (Arecaceae) with comparisons of supertree and supermatrix approaches. Systematic Biology, 58, 240–256.

Bouchenak2010 BOUCHENAK-KHELLADI, Y., VERBOOM, G. A., SAVOLAINEN, V. and HODKINSON, T. R. 2010. Biogeography of the grasses (Poaceae): a phylogenetic approach to reveal evolutionary history in geographical space and geological time. Botanical Journal of the Linnean Society, 162, 543–557.

Conrad2008 CONRAD, J. L. 2008. Phylogeny And Systematics Of Squamata (Reptilia) Based On Morphology. Bulletin of the American Museum of Natural History, 310, 1–182.

Dikow2009 DIKOW, T. 2009. A phylogenetic hypothesis for Asilidae based on a total evidence analysis of morphological and DNA sequence data (Insecta: Diptera: Brachycera: Asiloidea). Organisms Diversity and Evolution, 9, 165–188.

Eklund2004 EKLUND, H., DOYLE, J. A. and HERENDEEN, P. S. 2004. Morphological phylogenetic analysis of living and fossil Chloranthaceae. International Journal of Plant Sciences, 165, 107–151.

Geisler2001 GEISLER, J. H. 2001. New morphological evidence for the phylogeny of Artiodactyla, Cetacea, and Mesonychidae. American Museum Novitates, 3344, 53.

Giles2015 GILES, S., FRIEDMAN, M. and BRAZEAU, M. D. 2015. Osteichthyan-like cranial conditions in an Early Devonian stem gnathostome. Nature, 520, 82–85.

Griswold1999 GRISWOLD, C. E., CODDINGTON, J. A., PLATNICK, N. I. and FORSTER, R. R. 1999. Towards a phylogeny of entelegyne spiders (Araneae, Araneomorphae, Entelegynae). Journal of Arachnology, 27, 53–63.

Liljeblad2008 LILJEBLAD, J., RONQUIST, F., NIEVES-ALDREY, J. L., FONTAL-CAZALLA, F., ROS-FARRE, P., GAITROS, D. and PUJADE-VILLAR, J. 2008. A fully web-illustrated morphological phylogenetic study of relationships among oak gall wasps and their closest relatives (Hymenoptera: Cynipidae).

Loconte1991 LOCONTE, H. and STEVENSON, D. W. 1991. Cladistics of the Magnoliidae. Cladistics, 7, 267–296.

Longrich2010 LONGRICH, N. R., SANKEY, J. and TANKE, D. 2010. *Texacephale langstoni*, a new genus of pachycephalosaurid (Dinosauria: Ornithischia) from the upper Campanian Aguja Formation, southern Texas, USA. Cretaceous Research, 31, 274–284.

OMeara2014 O'MEARA, R. N. and THOMPSON, R. S. 2014. Were There Miocene Meridiolestidans? Assessing the phylogenetic placement of *Necrolestes patagonensis* and the presence of a 40 million year Meridiolestidan ghost lineage. Journal of Mammalian Evolution, 21, 271–284.

Rougier2012 ROUGIER, G. W., WIBLE, J. R., BECK, R. M. D. and APESTEGUIA, S. 2012. The Miocene mammal *Necrolestes* demonstrates the survival of a Mesozoic nontherian lineage into the late Cenozoic of South America. Proceedings of the National Academy of Sciences, 109, 20053–8.

Sharkey2011 SHARKEY, M. J., CARPENTER, J. M., VILHELMSEN, L., HERATY, J., LILJEBLAD, J., DOWLING, A. P. G., SCHULMEISTER, S., MURRAY, D., DEANS, A. R., RONQUIST, F., KROGMANN, L. and WHEELER, W. C. 2012. Phylogenetic relationships among superfamilies of Hymenoptera. Cladistics, 28, 80–112.

Sundue2010 SUNDUE, M. A., ISLAM, M. B. and RANKER, T. A. 2010. Systematics of Grammitid Ferns (Polypodiaceae): Using Morphology and Plastid Sequence Data to Resolve the Circumscriptions of Melpomene and the Polyphyletic Genera *Lellingeria* and *Terpsichore*. Systematic Botany, 35, 701–715.

Vinther2008 VINTHER, J., VAN ROY, P. and BRIGGS, D. E. G. 2008. Machaeridians are Palaeozoic armoured annelids. Nature, 451, 185–188.

Wilson2003 WILSON, G. D. F. and EDGECOMBE, G. D. 2003. The Triassic isopod *Protamphisopus wianamattensis* (Chilton) and comparison by extant taxa (Crustacea, Phreatoicidea). Journal of Paleontology, 77, 454–470.

Wortley2006 WORTLEY, A. H. and SCOTLAND, R. W. 2006. The effect of combining molecular and morphological data in published phylogenetic analyses. Systematic Biology, 55, 677–685.

Zanol2014 ZANOL, J., HALANYCH, K. M. and FAUCHALD, K. 2014. Reconciling taxonomy and phylogeny in the bristleworm family Eunicidae (Polychaete, Annelida). Zoologica Scripta, 43, 79–100.

Zhu2013 ZHU, M., YU, X., AHLBERG, P. E., CHOO, B., LU, J., QIAO, T., QU, Q., ZHAO, W., JIA, L., BLOM, H. and ZHU, Y. 2013. A Silurian placoderm with osteichthyan-like marginal jaw bones. Nature, 502, 188–193.

References

Brazeau MD, Guillerme T, Smith MR (2019). "An algorithm for morphological phylogenetic analysis with inapplicable data." *Systematic Biology*, in press. doi: 10.1093/sysbio/syy083.

`inapplicable.datasets` *Thirty Datasets with Inapplicable data*

Description

These are the datasets used to evaluate the behaviour of the inapplicable algorithm in Brazeau, Guillerme and Smith (2017).

Usage

```
inapplicable.datasets
```

Format

An object of class `list` of length 30.

Details

The name of each item corresponds to the datasets listed below. Datasets are sorted into two subsets, each sorted alphabetically; the first subset comprise simpler datasets with faster processing times. The value is the dataset in the format generated by `read.nexus.data`.

Source

Agnarsson2004 AGNARSSON, I. 2004. Morphological phylogeny of cobweb spiders and their relatives (Araneae, Araneoidea, Theridiidae). Zoological Journal of the Linnean Society, 141, 447–626.

Capa2011 CAPA, M., HUTCHINGS, P., AGUADO, M. T. and BOTT, N. J. 2011. Phylogeny of Sabellidae (Annelida) and relationships with other taxa inferred from morphology and multiple genes. Cladistics, 27, 449–469.

DeAssis2011 DE ASSIS, J. E. and CHRISTOFFERSEN, M. L. 2011. Phylogenetic relationships within Maldanidae (Capitellida, Annelida), based on morphological characters. Systematics and Biodiversity, 9, 233–245.

OLeary1999 O'LEARY, M. A. and GEISLER, J. H. 1999. The position of Cetacea within Mammalia: phylogenetic analysis of morphological data from extinct and extant taxa. Systematic Biology, 48, 455–490.

Rousset2004 ROUSSET, V., ROUSE, G. W., SIDDALL, M. E., TILLIER, A. and PLEIJEL, F. 2004. The phylogenetic position of Siboglinidae (Annelida) inferred from 18S rRNA, 28S rRNA and morphological data. Cladistics, 20, 518–533.

Sano2011 SANO, M. and AKIMOTO, S.-I. 2011. Morphological phylogeny of gall-forming aphids of the tribe Eriosomatini (Aphididae: Eriosomatinae). Systematic Entomology, 36, 607–627.

Sansom2010 SANSOM, R. S., FREEDMAN, K., GABBOTT, S. E., ALDRIDGE, R. J. and PURNELL, M. A. 2010. Taphonomy and affinity of an enigmatic Silurian vertebrate, *Jamoytius kerwoodi* White. Palaeontology, 53, 1393–1409.

Schulze2007 SCHULZE, A., CUTLER, E. B. and GIRIBET, G. 2007. Phylogeny of sipunculan worms: A combined analysis of four gene regions and morphology. Molecular Phylogenetics and Evolution, 42, 171–92.

Shultz2007 SHULTZ, J. W. 2007. A phylogenetic analysis of the arachnid orders based on morphological characters. Zoological Journal of the Linnean Society, 150, 221–265.

Wetterer2000 WETTERER, A. L., ROCKKMAN, M. V. and SIMMONS, N. B. 2000. Phylogeny of phyllostomid bats (Mammalia: Chiroptera): data from diverse morphological systems, sex chromosomes, and restriction sites. Bulletin of the American Museum of Natural History, 248, 1–200.

Wills2012 WILLS, M. A., GERBER, S., RUTA, M. and HUGHES, M. 2012. The disparity of priapulid, archaeopriapulid and palaeoscolecid worms in the light of new data. Journal of Evolutionary Biology, 25, 2056–2076.

Aguado2009 AGUADO, M. T. and SAN MARTIN, G. 2009. Phylogeny of Syllidae (Polychaeta) based on morphological data. Zoologica Scripta, 38, 379–402.

Aria2015 ARIA, C., CARON, J. B. and GAINES, R. 2015. A large new leanchoiliid from the Burgess Shale and the influence of inapplicable states on stem arthropod phylogeny. Palaeontology, 58, 629–660.

Asher2005 ASHER, R. J. and HOFREITER, M. 2006. Tenrec phylogeny and the noninvasive extraction of nuclear DNA. Systematic biology, 55, 181–94.

Baker2009 BAKER, W. J., SAVOLAINEN, V., ASMUSSEN-LANGE, C. B., CHASE, M. W., DRANSFIELD, J., FOREST, F., HARLEY, M. M., UHL, N. W. and WILKINSON, M. 2009. Complete generic-level phylogenetic analyses of palms (Arecaceae) with comparisons of supertree and supermatrix approaches. Systematic Biology, 58, 240–256.

Bouchenak2010 BOUCHENAK-KHELLADI, Y., VERBOOM, G. A., SAVOLAINEN, V. and HODKINSON, T. R. 2010. Biogeography of the grasses (Poaceae): a phylogenetic approach to reveal evolutionary history in geographical space and geological time. Botanical Journal of the Linnean Society, 162, 543–557.

Conrad2008 CONRAD, J. L. 2008. Phylogeny And Systematics Of Squamata (Reptilia) Based On Morphology. Bulletin of the American Museum of Natural History, 310, 1–182.

Dikow2009 DIKOW, T. 2009. A phylogenetic hypothesis for Asilidae based on a total evidence analysis of morphological and DNA sequence data (Insecta: Diptera: Brachycera: Asiloidea). Organisms Diversity and Evolution, 9, 165–188.

Eklund2004 EKLUND, H., DOYLE, J. A. and HERENDEEN, P. S. 2004. Morphological phylogenetic analysis of living and fossil Chloranthaceae. International Journal of Plant Sciences, 165, 107–151.

Geisler2001 GEISLER, J. H. 2001. New morphological evidence for the phylogeny of Artiodactyla, Cetacea, and Mesonychidae. American Museum Novitates, 3344, 53.

Giles2015 GILES, S., FRIEDMAN, M. and BRAZEAU, M. D. 2015. Osteichthyan-like cranial conditions in an Early Devonian stem gnathostome. Nature, 520, 82–85.

Griswold1999 GRISWOLD, C. E., CODDINGTON, J. A., PLATNICK, N. I. and FORSTER, R. R. 1999. Towards a phylogeny of entelegyne spiders (Araneae, Araneomorphae, Entelegynae). Journal of Arachnology, 27, 53–63.

Liljeblad2008 LILJEBLAD, J., RONQUIST, F., NIEVES-ALDREY, J. L., FONTAL-CAZALLA, F., ROS-FARRE, P., GAITROS, D. and PUJADE-VILLAR, J. 2008. A fully web-illustrated morphological phylogenetic study of relationships among oak gall wasps and their closest relatives (Hymenoptera: Cynipidae).

Loconte1991 LOCONTE, H. and STEVENSON, D. W. 1991. Cladistics of the Magnoliidae. Cladistics, 7, 267–296.

Longrich2010 LONGRICH, N. R., SANKEY, J. and TANKE, D. 2010. *Texacephale langstoni*, a new genus of pachycephalosaurid (Dinosauria: Ornithischia) from the upper Campanian Aguja Formation, southern Texas, USA. Cretaceous Research, 31, 274–284.

OMeara2014 O'MEARA, R. N. and THOMPSON, R. S. 2014. Were There Miocene Meridiolestidans? Assessing the phylogenetic placement of *Necrolestes patagonensis* and the presence of a 40 million year Meridiolestidan ghost lineage. Journal of Mammalian Evolution, 21, 271–284.

Rougier2012 ROUGIER, G. W., WIBLE, J. R., BECK, R. M. D. and APESTEGUIA, S. 2012. The Miocene mammal *Necrolestes* demonstrates the survival of a Mesozoic nontherian lineage into the late Cenozoic of South America. Proceedings of the National Academy of Sciences, 109, 20053–8.

Sharkey2011 SHARKEY, M. J., CARPENTER, J. M., VILHELMSEN, L., HERATY, J., LILJEBLAD, J., DOWLING, A. P. G., SCHULMEISTER, S., MURRAY, D., DEANS, A. R., RONQUIST, F., KROGMANN, L. and WHEELER, W. C. 2012. Phylogenetic relationships among superfamilies of Hymenoptera. Cladistics, 28, 80–112.

Sundue2010 SUNDUE, M. A., ISLAM, M. B. and RANKER, T. A. 2010. Systematics of Grammitid Ferns (Polypodiaceae): Using Morphology and Plastid Sequence Data to Resolve the Circumscriptions of Melpomene and the Polyphyletic Genera *Lellingeria* and *Terpsichore*. Systematic Botany, 35, 701–715.

Vinther2008 VINTHER, J., VAN ROY, P. and BRIGGS, D. E. G. 2008. Machaeridians are Palaeozoic armoured annelids. Nature, 451, 185–188.

Wilson2003 WILSON, G. D. F. and EDGECOMBE, G. D. 2003. The Triassic isopod *Protamphisopus wianamattensis* (Chilton) and comparison by extant taxa (Crustacea, Phreatoicidea). Journal of Paleontology, 77, 454–470.

Wortley2006 WORTLEY, A. H. and SCOTLAND, R. W. 2006. The effect of combining molecular and morphological data in published phylogenetic analyses. Systematic Biology, 55, 677–685.

Zanol2014 ZANOL, J., HALANYCH, K. M. and FAUCHALD, K. 2014. Reconciling taxonomy and phylogeny in the bristleworm family Eunicidae (Polychaete, Annelida). Zoologica Scripta, 43, 79–100.

Zhu2013 ZHU, M., YU, X., AHLBERG, P. E., CHOO, B., LU, J., QIAO, T., QU, Q., ZHAO, W., JIA, L., BLOM, H. and ZHU, Y. 2013. A Silurian placoderm with osteichthyan-like marginal jaw bones. Nature, 502, 188–193.

References

Brazeau MD, Guillerme T, Smith MR (2019). "An algorithm for morphological phylogenetic analysis with inapplicable data." *Systematic Biology*, in press. doi: 10.1093/sysbio/syy083.

`inapplicable.phyData` *Thirty Datasets with Inapplicable data*

Description

These are the datasets used to evaluate the behaviour of the inapplicable algorithm in Brazeau, Guillerme and Smith (2017).

Usage

```
inapplicable.phyData
```

Format

An object of class `list` of length 30.

Details

The name of each item corresponds to the datasets listed below. Datasets are sorted into two subsets, each sorted alphabetically; the first subset comprise simpler datasets with faster processing times. The value is the dataset in `phyDat` format.

Source

Agnarsson2004 AGNARSSON, I. 2004. Morphological phylogeny of cobweb spiders and their relatives (Araneae, Araneoidea, Theridiidae). Zoological Journal of the Linnean Society, 141, 447–626.

Capa2011 CAPA, M., HUTCHINGS, P., AGUADO, M. T. and BOTT, N. J. 2011. Phylogeny of Sabellidae (Annelida) and relationships with other taxa inferred from morphology and multiple genes. Cladistics, 27, 449–469.

DeAssis2011 DE ASSIS, J. E. and CHRISTOFFERSEN, M. L. 2011. Phylogenetic relationships within Maldanidae (Capitellida, Annelida), based on morphological characters. Systematics and Biodiversity, 9, 233–245.

OLeary1999 O'LEARY, M. A. and GEISLER, J. H. 1999. The position of Cetacea within Mammalia: phylogenetic analysis of morphological data from extinct and extant taxa. Systematic Biology, 48, 455–490.

Rousset2004 ROUSSET, V., ROUSE, G. W., SIDDALL, M. E., TILLIER, A. and PLEIJEL, F. 2004. The phylogenetic position of Siboglinidae (Annelida) inferred from 18S rRNA, 28S rRNA and morphological data. Cladistics, 20, 518–533.

Sano2011 SANO, M. and AKIMOTO, S.-I. 2011. Morphological phylogeny of gall-forming aphids of the tribe Eriosomatini (Aphididae: Eriosomatinae). Systematic Entomology, 36, 607–627.

Sansom2010 SANSOM, R. S., FREEDMAN, K., GABBOTT, S. E., ALDRIDGE, R. J. and PURNELL, M. A. 2010. Taphonomy and affinity of an enigmatic Silurian vertebrate, *Jamoytius kerwoodi* White. Palaeontology, 53, 1393–1409.

Schulze2007 SCHULZE, A., CUTLER, E. B. and GIRIBET, G. 2007. Phylogeny of sipunculan worms: A combined analysis of four gene regions and morphology. Molecular Phylogenetics and Evolution, 42, 171–92.

Shultz2007 SHULTZ, J. W. 2007. A phylogenetic analysis of the arachnid orders based on morphological characters. Zoological Journal of the Linnean Society, 150, 221–265.

Wetterer2000 WETTERER, A. L., ROCKKMAN, M. V. and SIMMONS, N. B. 2000. Phylogeny of phyllostomid bats (Mammalia: Chiroptera): data from diverse morphological systems, sex chromosomes, and restriction sites. Bulletin of the American Museum of Natural History, 248, 1–200.

Wills2012 WILLS, M. A., GERBER, S., RUTA, M. and HUGHES, M. 2012. The disparity of priapulid, archaeopriapulid and palaeoscolecid worms in the light of new data. Journal of Evolutionary Biology, 25, 2056–2076.

Aguado2009 AGUADO, M. T. and SAN MARTIN, G. 2009. Phylogeny of Syllidae (Polychaeta) based on morphological data. Zoologica Scripta, 38, 379–402.

Aria2015 ARIA, C., CARON, J. B. and GAINES, R. 2015. A large new leanchoiliid from the Burgess Shale and the influence of inapplicable states on stem arthropod phylogeny. Palaeontology, 58, 629–660.

Asher2005 ASHER, R. J. and HOFREITER, M. 2006. Tenrec phylogeny and the noninvasive extraction of nuclear DNA. Systematic biology, 55, 181–94.

Baker2009 BAKER, W. J., SAVOLAINEN, V., ASMUSSEN-LANGE, C. B., CHASE, M. W., DRANSFIELD, J., FOREST, F., HARLEY, M. M., UHL, N. W. and WILKINSON, M. 2009. Complete generic-level phylogenetic analyses of palms (Arecaceae) with comparisons of supertree and supermatrix approaches. Systematic Biology, 58, 240–256.

Bouchenak2010 BOUCHENAK-KHELLADI, Y., VERBOOM, G. A., SAVOLAINEN, V. and HODKINSON, T. R. 2010. Biogeography of the grasses (Poaceae): a phylogenetic approach to reveal evolutionary history in geographical space and geological time. Botanical Journal of the Linnean Society, 162, 543–557.

Conrad2008 CONRAD, J. L. 2008. Phylogeny And Systematics Of Squamata (Reptilia) Based On Morphology. Bulletin of the American Museum of Natural History, 310, 1–182.

Dikow2009 DIKOW, T. 2009. A phylogenetic hypothesis for Asilidae based on a total evidence analysis of morphological and DNA sequence data (Insecta: Diptera: Brachycera: Asiloidea). Organisms Diversity and Evolution, 9, 165–188.

Eklund2004 EKLUND, H., DOYLE, J. A. and HERENDEEN, P. S. 2004. Morphological phylogenetic analysis of living and fossil Chloranthaceae. International Journal of Plant Sciences, 165, 107–151.

Geisler2001 GEISLER, J. H. 2001. New morphological evidence for the phylogeny of Artiodactyla, Cetacea, and Mesonychidae. American Museum Novitates, 3344, 53.

Giles2015 GILES, S., FRIEDMAN, M. and BRAZEAU, M. D. 2015. Osteichthyan-like cranial conditions in an Early Devonian stem gnathostome. Nature, 520, 82–85.

Griswold1999 GRISWOLD, C. E., CODDINGTON, J. A., PLATNICK, N. I. and FORSTER, R. R. 1999. Towards a phylogeny of entelegyne spiders (Araneae, Araneomorphae, Entelegynae). Journal of Arachnology, 27, 53–63.

Liljeblad2008 LILJEBLAD, J., RONQUIST, F., NIEVES-ALDREY, J. L., FONTAL-CAZALLA, F., ROS-FARRE, P., GAITROS, D. and PUJADE-VILLAR, J. 2008. A fully web-illustrated morphological phylogenetic study of relationships among oak gall wasps and their closest relatives (Hymenoptera: Cynipidae).

Loconte1991 LOCONTE, H. and STEVENSON, D. W. 1991. Cladistics of the Magnoliidae. Cladistics, 7, 267–296.

Longrich2010 LONGRICH, N. R., SANKEY, J. and TANKE, D. 2010. *Texacephale langstoni*, a new genus of pachycephalosaurid (Dinosauria: Ornithischia) from the upper Campanian Aguja Formation, southern Texas, USA. Cretaceous Research, 31, 274–284.

OMeara2014 O'MEARA, R. N. and THOMPSON, R. S. 2014. Were There Miocene Meridiolestidans? Assessing the phylogenetic placement of *Necrolestes patagonensis* and the presence of a 40 million year Meridiolestidan ghost lineage. Journal of Mammalian Evolution, 21, 271–284.

Rougier2012 ROUGIER, G. W., WIBLE, J. R., BECK, R. M. D. and APESTEGUIA, S. 2012. The Miocene mammal *Necrolestes* demonstrates the survival of a Mesozoic nontherian lineage into the late Cenozoic of South America. Proceedings of the National Academy of Sciences, 109, 20053–8.

Sharkey2011 SHARKEY, M. J., CARPENTER, J. M., VILHELMSEN, L., HERATY, J., LILJEBLAD, J., DOWLING, A. P. G., SCHULMEISTER, S., MURRAY, D., DEANS, A. R., RONQUIST, F., KROGMANN, L. and WHEELER, W. C. 2012. Phylogenetic relationships among superfamilies of Hymenoptera. Cladistics, 28, 80–112.

Sundue2010 SUNDUE, M. A., ISLAM, M. B. and RANKER, T. A. 2010. Systematics of Grammitid Ferns (Polypodiaceae): Using Morphology and Plastid Sequence Data to Resolve the Circumscriptions of Melpomene and the Polyphyletic Genera *Lellingeria* and *Terpsichore*. Systematic Botany, 35, 701–715.

Vinther2008 VINTHER, J., VAN ROY, P. and BRIGGS, D. E. G. 2008. Machaeridians are Palaeozoic armoured annelids. Nature, 451, 185–188.

Wilson2003 WILSON, G. D. F. and EDGECOMBE, G. D. 2003. The Triassic isopod *Protamphisopus wianamattensis* (Chilton) and comparison by extant taxa (Crustacea, Phreatoicidea). Journal of Paleontology, 77, 454–470.

Wortley2006 WORTLEY, A. H. and SCOTLAND, R. W. 2006. The effect of combining molecular and morphological data in published phylogenetic analyses. Systematic Biology, 55, 677–685.

Zanol2014 ZANOL, J., HALANYCH, K. M. and FAUCHALD, K. 2014. Reconciling taxonomy and phylogeny in the bristleworm family Eunicidae (Polychaete, Annelida). Zoologica Scripta, 43, 79–100.

Zhu2013 ZHU, M., YU, X., AHLBERG, P. E., CHOO, B., LU, J., QIAO, T., QU, Q., ZHAO, W., JIA, L., BLOM, H. and ZHU, Y. 2013. A Silurian placoderm with osteichthyan-like marginal jaw bones. Nature, 502, 188–193.

References

Brazeau MD, Guillerme T, Smith MR (2019). "An algorithm for morphological phylogenetic analysis with inapplicable data." *Systematic Biology*, in press. doi: 10.1093/sysbio/syy083.

`InfoAmounts` *Amount of information in each character*

Description

As presently implemented, this function requires that there be no ambiguous tokens and two applicable tokens, '1' and '2'.

Usage

```
InfoAmounts(tokenTable, precision = 1e+06, warn = TRUE)
```

Arguments

`tokenTable`	A matrix, where each row corresponds to a character, each column to a tip, and each entry to the value (1 or 2) of the character at that tip.
`precision`	number of random trees to generate when calculating Profile curves
`warn`	Boolean (`TRUE`/`FALSE`): display warnings when concavity functions are generated by approximation.

Value

information content of each extra step, in bits

Author(s)

Martin R. Smith

IWScore *Implied weights parsimony Score*

Description

Calculate a tree's Parsimony score with a given dataset using implied weights (Goloboff 1997)

Usage

```
IWScore(tree, dataset, concavity = 4, ...)

IWInitMorphy(dataset)

IWDestroyMorphy(dataset)
```

Arguments

tree	A tree of class phylo.
dataset	Dataset of class phyDat. The dataset should have been prepared using dataset <- PrepareDataIW(data if this step has not been completed, the dataset will be (time-consumingly) prepared within the function. In subsidiary functions, the dataset will have been initialized using IWInitMorphy, and must be destroyed using IWDestroyMorphy.
concavity	A numeric value to use as the concavity constant (k) in implied weighting.
...	unused; allows additional parameters specified within ... to be received by the function without throwing an error.

Value

The 'fit', h / h + k, where h is the amount of homoplasy ('extra steps') and k is a constant (the 'concavity constant')

Functions

- IWInitMorphy: Initialize dataset by adding morphyObjs and min.steps.
- IWDestroyMorphy: Free memory from morphyObjs initialized by IWScoreMorphy.

Author(s)

Martin R. Smith

References

Goloboff PA (1997). "Self-weighted optimization: tree searches and character state reconstructions under implied transformation costs." *Cladistics*, **13**(3), 225–245. http://dx.doi.org/10.1111/j.1096-0031.1997.tb00317.x.

Examples

```
data(referenceTree)
data(congreveLamsdellMatrices)
dataset <- PrepareDataIW(congreveLamsdellMatrices[[42]])
IWScore(referenceTree, dataset)
```

`IWScoreMorphy` *Profile Parsimony Score*

Description

Calculate a tree's Profile Parsimony score with a given dataset, after Faith and Trueman (2001)

Usage

```
IWScoreMorphy(parent, child, dataset, concavity = 4,
  minSteps = attr(dataset, "min.steps"), ...)

ProfileScore(tree, dataset)

ProfileScoreMorphy(parent, child, dataset, ...)

ProfileInitMorphy(dataset)

ProfileDestroyMorphy(dataset)
```

Arguments

`parent`	the first column of the edge matrix of a tree of class `phylo`, i.e. `tree$edge[, 1]`
`child`	the second column of the edge matrix of a tree of class `phylo`, i.e. `tree$edge[, 2]`
`dataset`	Dataset of class `phyDat`. The dataset should have been prepared using `dataset <- PrepareDataProfile` if this step has not been completed, the dataset will be (time-consumingly) prepared within the function. In subsidiary functions, the dataset will have been initialized using `ProfileInitMorphy`, must be destroyed using `ProfileDestroyMorphy`.
`concavity`	A numeric value to use as the concavity constant (k) in implied weighting.
`minSteps`	Integer vector specifying the minimum number of steps possible for each character in `dataset`, perhaps calculated using `MinimumSteps`.
`...`	unused; allows additional parameters specified within ... to be received by the function without throwing an error.
`tree`	A tree of class `phylo`.

Value

Zero minus the profile score (because the optimization algorithm treats smaller numbers as better)

Functions

- `IWScoreMorphy`: Scorer for initialized dataset.
- `ProfileScoreMorphy`: Scorer for initialized dataset.
- `ProfileInitMorphy`: Initialize dataset by adding morphyObjs.
- `ProfileDestroyMorphy`: Free memory from morphyObjs initialized by `ProfileScoreMorphy`.

Author(s)

Martin R. Smith

References

Faith DP, Trueman JWH (2001). "Towards an inclusive philosophy for phylogenetic inference." *Systematic Biology*, **50**(3), 331–350. doi: 10.1080/106351501186627.

Examples

```
data(referenceTree)
data(congreveLamsdellMatrices)
# In actual use, the dataset should be prepared with a much higher
# precision: try 1e+06?
# Of course, gaining higher precision takes substantially longer.
dataset <- PrepareDataProfile(congreveLamsdellMatrices[[42]], precision=1e+03)
ProfileScore(referenceTree, dataset)
```

`JointInformation` *Joint Information of two splits*

Description

Calculates the joint phylogenetic information content of two splits: that is, the information content of the two splits considered separately, minus their mutual information (which would otherwise be counted twice).

Usage

```
JointInformation(A1A2, A1B2, B1A2, B1B2)
```

Arguments

`A1A2, A1B2, B1A2, B1B2`
: Number of taxa common to splits A1 and A2 (etc.).

Details

Because some information is common to both splits (`SplitMutualInformation`), the joint information of two splits will be less than the sum of the information of the splits taken separately – unless the splits are contradictory.

Split Y1 is defined as dividing taxa into the two sets A1 and B1, and Y2=A2:B2.

Consider partitions that divide eight terminals, labelled A to H.

Bipartition 1:	ABCD:EFGH	A1 = ABCD	B1 = EFGH
Bipartition 2:	ABC:DEFGH	A2 = ABC	B2 = DEFGH

This can be represented by an association matrix:

	A2	*B2*
A1	ABC	D
B1		EFGH

The joint information is given by `JointInformation(3, 1, 0, 4)`.

Value

`JointInformation` returns the joint phylogenetic information content of two splits, measured in bits.

Author(s)

Martin R. Smith

Examples

```
JointInformation(3, 1, 0, 4)
```

`Lobo.data` *Raw data from Zhang et al. 2016*

Description

Raw data from Zhang et al. 2016

Modified so that absences are treated appropriately:

Character 7 inapplicable to absent where cephalic shield (char 3) is absent

Character 40 inapplicable to absent where paired appendages absent

Character 46 inapplicable to absent

Character 64 76: stet; trunk annulations / limbs may primitively have been papillate or non-papillate

Character 69 a good case that the fusion of flaps with endopods is secondary; thus inapplicable to absent where 67 is applicable

Character 72 inapplicable to absent where appendages are present

Character 77 inapplicable to absent where papillae applicable, as spine is a secondary elaboration of papillae

Character 78 inapplicable to absent

Character 79 stet, as possible that limbs evolved by extension of plate-like exoskeletal element

Character 80 inapplicable to absent, on assumption that ancestral claws were simple

Characters 83, 84, 86, 87, 92 inapplicable to absent; an obvious elaboration

Character 93 inapplicable to absent; ancestrally undifferentiated by definition?

Character 96 inapplicable to absent; ancestrally undifferentiated by definition. Ambiguous in taxa that lack claws as rotation would not be observed.]

Usage

```
Lobo.data
```

Format

An object of class `list` of length 48.

Source

Zhang X, Smith MR, Yang J, Hou J (2016). "Onychophoran-like musculature in a phosphatized Cambrian lobopodian." *Biology Letters*, **12**(9), 20160492. doi: 10.1098/rsbl.2016.0492, http://rsbl.royalsocietypublishing.org/content/12/9/20160492.

`Lobo.phy` *Data from Zhang et al. 2016 in phyDat format*

Description

Data from Zhang et al. 2016 in phyDat format

Modified so that absences are treated appropriately:

Character 7 inapplicable to absent where cephalic shield (char 3) is absent

Character 40 inapplicable to absent where paired appendages absent

Character 46 inapplicable to absent

Character 64 76: stet; trunk annulations / limbs may primitively have been papillate or non-papillate

Character 69 a good case that the fusion of flaps with endopods is secondary; thus inapplicable to absent where 67 is applicable

Character 72 inapplicable to absent where appendages are present

Character 77 inapplicable to absent where papillae applicable, as spine is a secondary elaboration of papillae

Character 78 inapplicable to absent

Character 79 stet, as possible that limbs evolved by extension of plate-like exoskeletal element

Character 80 inapplicable to absent, on assumption that ancestral claws were simple

Characters 83, 84, 86, 87, 92 inapplicable to absent; an obvious elaboration

Character 93 inapplicable to absent; ancestrally undifferentiated by definition?

Character 96 inapplicable to absent; ancestrally undifferentiated by definition. Ambiguous in taxa that lack claws as rotation would not be observed.]

Usage

```
Lobo.phy
```

Format

An object of class `phyDat` of length 48.

Source

Zhang X, Smith MR, Yang J, Hou J (2016). "Onychophoran-like musculature in a phosphatized Cambrian lobopodian." *Biology Letters*, **12**(9), 20160492. doi: 10.1098/rsbl.2016.0492, http://rsbl.royalsocietypublishing.org/content/12/9/20160492.

---

logDoubleFactorials *Natural logarithms of double factorials*

Description

A vector with pre-calculated values of double factorials up to 50 000!!.

Usage

```
logDoubleFactorials
```

Format

An object of class `numeric` of length 50000.

LogisticPoints *Logistic Points Extract points from a fitted model*

Description

Logistic Points Extract points from a fitted model

Usage

```
LogisticPoints(x, fittedModel)
```

Arguments

x	an integer vector giving x co-ordinates.
fittedModel	a fitted model, perhaps generated using `nls(cumP ~ SSlogis(nSteps, Asym, xmid, scal))`.

Value

values of y co-ordinates corresponding to the x co-ordinates provided

Author(s)

Martin R. Smith

MatchingSplitDistance *Matching Split Distance*

Description

Implements the Matching Split Distance for unrooted binary phylogenetic trees of Bogdanowicz and Giaro (2012).

Usage

```
MatchingSplitDistance(tree1, tree2, reportMatching = FALSE)

MatchingSplitDistanceSplits(splits1, splits2, reportMatching = FALSE)
```

Arguments

tree1	Trees of class phylo, with tips labelled identically, or lists of such trees to undergo pairwise comparison.
tree2	Trees of class phylo, with tips labelled identically, or lists of such trees to undergo pairwise comparison.
reportMatching	Logical specifying whether to return the clade matchings as an attribute of the score.
splits1	Logical matrices where each row corresponds to a terminal, either listed in the same order or bearing identical names (in any sequence), and each column corresponds to a bipartition split, such that each terminal is identified as a member of the ingroup (TRUE) or outgroup (FALSE) of the respective bipartition split.
splits2	Logical matrices where each row corresponds to a terminal, either listed in the same order or bearing identical names (in any sequence), and each column corresponds to a bipartition split, such that each terminal is identified as a member of the ingroup (TRUE) or outgroup (FALSE) of the respective bipartition split.

Functions

- MatchingSplitDistanceSplits: Takes splits instead of trees

Author(s)

Martin R. Smith

References

Bogdanowicz D, Giaro K (2012). "Matching split distance for unrooted binary phylogenetic trees." *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, **9**(1), 150–160. doi: 10.1109/TCBB.2011.48.

MatrixToList *Edge matrix to edge list*

Description

Edge matrix to edge list

Usage

```
MatrixToList(edge)
```

Arguments

edge	edges in the matrix format used by tree$edge, where tree is a tree of class phylo

Value

tree edges in the format list(parent, child).

`MinimumSteps` *Minimum steps*

Description

The smallest number of steps that a character can take on any tree.

Usage

```
MinimumSteps(states)
```

Arguments

`states`	Integer vector listing the tokens that may be present at each tip along a single character, with each token represented as a binary digit; e.g. a value of 11 means that the tip may have tokens 0, 1 or 3 (as 11 = 2^0 + 2^1 + 2^3). As the minimum steps can be found when inapplicables occur together, inapplicable tokens can be denoted as ?s or with the integer 0 (not 2^0). Tokens that are ambiguous for an inapplicable and an applicable state are not presently supported.

Value

An integer specifying the minimum number of steps that the character must contain.

Author(s)

Martin R. Smith

Examples

```
{
  data('inapplicable.datasets')
  myPhyDat <- inapplicable.phyData[[4]] # or as.phyDat(read.nexus.data('filepath'))
  class(myPhyDat) # phyDat object

  # Convert list of character codings to an array
  myData <- vapply(myPhyDat, I, myPhyDat[[1]])

  # Convert phyDat's representation of states to binary
  myContrast <- attr(myPhyDat, 'contrast')
  tokens <- colnames(myContrast)
  binaryContrast <- integer(length(tokens))
  tokenApplicable <- tokens != '-'
  binaryContrast[tokenApplicable] <- 2 ^ (seq_len(sum(tokenApplicable)) - 1)
  binaryValues <- apply(myContrast, 1,
    function (row) sum(binaryContrast[as.logical(row)]))
```

```
  myStates <- matrix(binaryValues[myData], nrow=nrow(myData),
                     ncol=ncol(myData), dimnames=dimnames(myData))

  # Finally, work out minimum steps
  apply(myStates, 1, MinimumSteps)

}
```

MorphyBootstrap *Ratchet bootstrapper*

Description

Ratchet bootstrapper

Usage

```
MorphyBootstrap(edgeList, morphyObj, EdgeSwapper = NNISwap, maxIter,
  maxHits, verbosity = 1L, stopAtPeak = FALSE, stopAtPlateau = 0L,
  ...)

ProfileBootstrap(edgeList, dataset, EdgeSwapper = NNISwap, maxIter,
  maxHits, verbosity = 1L, ...)

IWBootstrap(edgeList, dataset, concavity = 4L, EdgeSwapper = NNISwap,
  maxIter, maxHits, verbosity = 1L, ...)
```

Arguments

edgeList	a list containing the following: - vector of integers corresponding to the parent of each edge in turn - vector of integers corresponding to the child of each edge in turn - (optionally) score of the tree - (optionally, if score provided) number of times this score has been hit
morphyObj	A morphy object, perhaps created with PhyDat2Morphy.
EdgeSwapper	a function that rearranges a parent and child vector, and returns a list with modified vectors; for example SPRSwap.
maxIter	maximum number of iterations to perform in tree search
maxHits	maximum number of hits to accomplish in tree search
verbosity	Level of detail to display in console: larger numbers provide more verbose feedback to the user.
stopAtPeak	Logical specifying whether to terminate search once a subsequent iteration recovers a sub-optimal score. Useful with methods that return all trees one rearrangement from the current tree, such as AllTBR. Will be overridden if a passed function has an attribute stopAtPeak set by attr(FunctionName, 'stopAtPeak') <- TRUE.

stopAtPlateau	Integer. If > 0, tree search will terminate if the score has not improved after stopAtPlateau iterations. Useful with methods that return all trees one rearrangement from the current tree, such as AllTBR. Will be overridden if a passed function has an attribute stopAtPlateau set by attr(FunctionName, 'stopAtPlateau') <- TRUE.
...	further parameters to send to TreeScorer
dataset	A phylogenetic data matrix of class phyDat, whose names correspond to the labels of any accompanying tree.
concavity	A numeric value to use as the concavity constant (k) in implied weighting.

Value

A tree that is optimal under a random sampling of the original characters

Functions

- ProfileBootstrap: Bootstrapper for Profile Parsimony
- IWBootstrap: Bootstrapper for Implied weighting

MorphyWeights *Report the character weightings associated with a Morphy object*

Description

Report the character weightings associated with a Morphy object

Usage

```
MorphyWeights(morphyObj)
```

Arguments

morphyObj	A morphy object, perhaps created with PhyDat2Morphy.

Value

a matrix of dimensions (2, number of characters); row 1 lists the exact rates specified by the user; row 2 the approximate (integral) weights used by MorphyLib

Author(s)

Martin R. Smith

MutualArborealInfo *Information-based generalized Robinson-Foulds distance between two trees*

Description

Functions reporting the distances or similarities between pairs of trees, based on information-theoretic concepts.

Usage

```
MutualArborealInfo(tree1, tree2, reportMatching = FALSE)

VariationOfArborealInfo(tree1, tree2, reportMatching = FALSE)

MutualClusterInfo(tree1, tree2, reportMatching = FALSE,
  bestMatchOnly = TRUE)

MutualArborealInfoSplits(splits1, splits2, reportMatching = FALSE)

VariationOfArborealInfoSplits(splits1, splits2, reportMatching = FALSE)

MutualClusterInfoSplits(splits1, splits2, reportMatching = FALSE,
  bestMatchOnly = TRUE,
  partitionQualityIndex = SplitPairingInformationIndex(dim(splits1)[1]))
```

Arguments

tree1, tree2	Trees of class phylo, with tips labelled identically, or lists of such trees to undergo pairwise comparison.
reportMatching	Logical specifying whether to return the clade matchings as an attribute of the score.
bestMatchOnly	Logical specifying whether to return how informative each split is about its best match only (TRUE) or how informative each split is about each other split (FALSE).
splits1, splits2	Logical matrices where each row corresponds to a terminal, either listed in the same order or bearing identical names (in any sequence), and each column corresponds to a bipartition split, such that each terminal is identified as a member of the ingroup (TRUE) or outgroup (FALSE) of the respective bipartition split.
partitionQualityIndex	Output of SplitPairingInformationIndex for n taxa; calculated automatically if not specified, but passing a cached value may improve performance.

Details

Each partition in a tree can be viewed either as

- (a) a statement that the 'true' tree is one of those that splits the taxa as specified;
- (b) a statement that the taxa are subdivided into the two groups specified.

The former concept corresponds to the concept of phylogenetic information, and views the information content of a pair of partitions as relating to the proportion of phylogenetic trees that are consistent with both partitions, giving rise to the Mutual Arboreal Information similarity measure (`MutualArborealInfo`), and the complementary Variation of Arboreal Information distance metric (`VariationOfArborealInfo`).

The latter sees the information content of a pair of partitions as relating to the proportion of all possible pairings that are at least as similar (measured using Meila's (2007) the variation of information) as the pairing in question, giving rise to the Mutual Clustering Information similarity measure (`MutualClusterInfo`).

A tree similarity measure is generated by finding an optimal matching that maximises the total information in common between a partition on one tree and its pair on a second, considering all possible ways to pair partitions between trees (including leaving a partition unpaired).

The returned tree similarity measures state the amount of information, in bits, that the partitions in two trees hold in common when they are optimally matched, following Smith (forthcoming). The complementary tree distance measures state how much information is different in the partitions of two trees, under an optimal matching.

Value

If `reportMatching = FALSE`, the functions return a numeric vector specifying the requested similarities or differences.

If `reportMatching = TRUE`, the functions additionally return

Functions

- `VariationOfArborealInfo`: Variation of phylogenetic information between two trees
- `MutualClusterInfo`: Mutual clustering information between two trees
- `MutualArborealInfoSplits`: Takes splits instead of trees
- `VariationOfArborealInfoSplits`: Calculate variation of arboreal information from splits
- `MutualClusterInfoSplits`: Takes splits instead of trees

Author(s)

Martin R. Smith

References

- MeilÄ*f* M (2007). "Comparing clusterings—an information based distance." *Journal of Multivariate Analysis*, **98**(5), 873–895. doi: 10.1016/j.jmva.2006.11.013.
- Smith MR (2019). "Information theoretic Generalized Robinson-Foulds metrics for measuring the distance between phylogenetic trees." *Forthcoming*.

- Vinh NX, Epps J, Bailey J (2010). "Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance." *Journal of Machine Learning Research*, **11**, 2837–2854. doi: 10.1145/1553374.1553511.

Examples

```
{
  tree1 <- ape::read.tree(text='((((a, b), c), d), (e, (f, (g, h))));')
  tree2 <- ape::read.tree(text='(((a, b), (c, d)), ((e, f), (g, h)));')
  tree3 <- ape::read.tree(text='((((h, b), c), d), (e, (f, (g, a))));')

  # Best possible score is obtained by matching a tree with itself
  VariationOfArborealInfo(tree1, tree1) # 0, by definition
  MutualArborealInfo(tree1, tree1)

  # Best possible score is a function of tree shape; the partitions within
  # balanced trees are more independent and thus contain less information
  MutualArborealInfo(tree2, tree2)

  # How similar are two trees?
  MutualArborealInfo(tree1, tree2)
  VariationOfArborealInfo(tree1, tree2)
  VariationOfArborealInfo(tree2, tree1) # Identical, by symmetry


  # Maximum possible score for Cluster information is independent
  # of tree shape, as every possible pairing is considered
  MutualClusterInfo(tree1, tree1)
  MutualClusterInfo(tree2, tree2)

  # It is thus easier to interpret the value of
  MutualClusterInfo(tree1, tree2)
  # Although it may not be possible to find a tree pair with zero mutual
  # cluster info.

  # Every partition in tree1 is contradicted by every partition in tree3
  # Non-arboreal matches contain clustering, but not phylogenetic, information
  MutualArborealInfo(tree1, tree3) # = 0
  MutualClusterInfo(tree1, tree3) # > 0

}
```

N1Spr *Number of trees one SPR step away Formula given by Allen and Steel (2001).*

Description

Number of trees one SPR step away Formula given by Allen and Steel (2001).

Usage

```
N1Spr(n)

IC1Spr(n)
```

Arguments

n	Number of tips in tree.

Functions

- IC1Spr: Information content of trees 0 or 1 SPR step from tree with n tips.

References

Allen BL, Steel MA (2001). "Subtree transfer operations and their induced metrics on evolutionary trees." *Annals of Combinatorics*, **5**(1), 1–15. ISSN 0218-0006, doi: 10.1007/s0002600180068.

NewickTree *Newick Tree*

Description

Writes a tree in Newick format

Usage

```
NewickTree(tree)
```

Arguments

tree	A tree of class phylo.

Value

A character string describing tree in Newick format

NJTree *Neighbour Joining Tree*

Description

Generates a rooted neighbour joining tree, with no edge lengths

Usage

```
NJTree(dataset)
```

Arguments

dataset	A phylogenetic data matrix of class phyDat, whose names correspond to the labels of any accompanying tree.

Value

an object of class phylo

Author(s)

Martin R. Smith

NNI *Nearest Neighbour Interchange (NNI)*

Description

Performs a single iteration of the nearest-neighbour interchange algorithm. Based on the corresponding phangorn function, but re-coded to improve speed.

Usage

```
NNI(tree, edgeToBreak = NULL)

NNISwap(parent, child, nTips = (length(parent)/2L) + 1L,
  edgeToBreak = NULL)

RootedNNI(tree, edgeToBreak = NULL)

RootedNNISwap(parent, child, nTips = (length(parent)/2L) + 1L,
  edgeToBreak = NULL)
```

Arguments

tree	A tree of class phylo.
edgeToBreak	(optional) integer specifying the index of an edge to bisect/prune, generated randomly if not specified. Alternatively, set to -1 to return a complete list of all trees one step from the input tree.
parent	the first column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 1]
child	the second column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 2]
nTips	(optional) Number of tips.

Details

Branch lengths are not supported.

All nodes in a tree must be bifurcating; ape::collapse.singles and ape::multi2di may help.

Value

Returns a tree with class phylo (if returnAll = FALSE) or a set of trees, with class multiPhylo (if returnAll = TRUE).

a list containing two elements, corresponding in turn to the rearranged parent and child parameters

a list containing two elements, corresponding in turn to the rearranged parent and child parameters

Functions

- NNISwap: faster version that takes and returns parent and child parameters
- RootedNNI: Perform NNI rearrangement, retaining position of root
- RootedNNISwap: faster version that takes and returns parent and child parameters

Author(s)

Martin R. Smith

References

The algorithm is summarized in Felsenstein J (2004). *Inferring phylogenies*. Sinauer Associates, Sunderland, Massachusetts.

Examples

```
tree <- ape::rtree(20, br=NULL)
NNI(tree)
NNI(tree, edgeToBreak = -1)
```

`NPartitionPairs` *Distributions of taxa consistent with a partition pair.*

Description

Number of terminal arrangements matching a specified configuration of two partitions.

Usage

```
NPartitionPairs(configuration)
```

Arguments

`configuration`	Integer vector of length four specifying the number of terminals that occur in both (1) splits A1 and A2; (2) splits A1 and B2; (3) splits B1 and A2; (4) splits B1 and B2.

Details

Consider partitions that divide eight terminals, labelled A to H.

Bipartition 1:	ABCD:EFGH	A1 = ABCD	B1 = EFGH
Bipartition 2:	ABE:CDFGH	A2 = ABE	B2 = CDFGH

This can be represented by an association matrix:

	A2	*B2*
A1	AB	C
B1	E	FGH

The cells in this matrix contain 2, 1, 1 and 3 terminals respectively; this four-element vector (`c(2, 1, 1, 3)`) is the `configuration` implied by this pair of bipartition splits.

Value

The number of ways to distribute `sum(configuration)` taxa according to the specified pattern.

Author(s)

Martin R. Smith

Examples

```
NPartitionPairs(c(2, 1, 1, 3))
```

NRooted *Number of rooted/unrooted trees These functions return the number of rooted or unrooted trees consistent with a given pattern of splits.*

Description

Functions starting N return the number of rooted or unrooted trees, functions starting Ln provide the log of this number. Calculations follow Carter et al. 1990, Theorem 2.

Usage

```
NRooted(tips)

NUnrooted(tips)

LnUnrooted(tips)

LnRooted(tips)

LnUnrootedSplits(splits)

NUnrootedSplits(splits)

LnUnrootedMult(splits)

NUnrootedMult(splits)
```

Arguments

tips	The number of tips.
splits	vector listing the number of taxa in each tree bipartition.

Functions

- NUnrooted: Number of unrooted trees
- LnUnrooted: Log Number of unrooted trees
- LnRooted: Log Number of rooted trees
- LnUnrootedSplits: Log number of unrooted trees
- NUnrootedSplits: Number of unrooted trees
- LnUnrootedMult: Log unrooted mult
- NUnrootedMult: Number of unrooted trees (mult)

Author(s)

Martin R. Smith

References

Carter M, Hendy M, Penny D, SzÃ©kely LA, Wormald NC (1990). "On the distribution of lengths of evolutionary trees." *SIAM Journal on Discrete Mathematics*, **3**(1), 38–47. doi: 10.1137/0403005, http://epubs.siam.org/doi/abs/10.1137/0403005.

Carter M, Hendy M, Penny D, SzÃ©kely LA, Wormald NC (1990). "On the distribution of lengths of evolutionary trees." *SIAM Journal on Discrete Mathematics*, **3**(1), 38–47. doi: 10.1137/0403005, http://epubs.siam.org/doi/abs/10.1137/0403005.

Examples

```
NRooted(10)
NUnrooted(10)
LnRooted(10)
LnUnrooted(10)
# Number of trees consistent with a character whose states are 00000 11111 222
NUnrootedMult(c(5,5,3))
```

`NyeTreeSimilarity` *Nye et al. (2006) tree comparison*

Description

Implements the tree comparison metric of Nye *et al.* (2006).

Usage

```
NyeTreeSimilarity(tree1, tree2, reportMatching = FALSE)

NyeSplitSimilarity(splits1, splits2, reportMatching = FALSE)
```

Arguments

`tree1`	Trees of class `phylo`, with tips labelled identically, or lists of such trees to undergo pairwise comparison.
`tree2`	Trees of class `phylo`, with tips labelled identically, or lists of such trees to undergo pairwise comparison.
`reportMatching`	Logical specifying whether to return the clade matchings as an attribute of the score.
`splits1`	Logical matrices where each row corresponds to a terminal, either listed in the same order or bearing identical names (in any sequence), and each column corresponds to a bipartition split, such that each terminal is identified as a member of the ingroup (`TRUE`) or outgroup (`FALSE`) of the respective bipartition split.
`splits2`	Logical matrices where each row corresponds to a terminal, either listed in the same order or bearing identical names (in any sequence), and each column corresponds to a bipartition split, such that each terminal is identified as a member of the ingroup (`TRUE`) or outgroup (`FALSE`) of the respective bipartition split.

Functions

- `NyeSplitSimilarity`: Takes splits instead of trees

Author(s)

Martin R. Smith

References

Nye TMW, LiÃ² P, Gilks WR (2006). "A novel algorithm and web-based tool for comparing two alternative phylogenetic trees." *Bioinformatics*, **22**(1), 117–119. doi: 10.1093/bioinformatics/bti720.

`PhyDat2Morphy` *Initialize a Morphy Object from a phyDat object*

Description

Creates a new Morphy object with the same size and characters as the phyDat object

Usage

```
PhyDat2Morphy(phy)
```

Arguments

`phy`	An object of class `phyDat`.

Value

A pointer to an initialized Morphy object.

Author(s)

Martin R. Smith

PhyToString *Extract character data from a phyDat object as a string*

Description

Extract character data from a phyDat object as a string

Usage

```
PhyToString(phy, ps = "", useIndex = TRUE, byTaxon = TRUE,
  concatenate = TRUE)
```

Arguments

phy	An object of class phyDat
ps	Character specifying text, perhaps ';', to append to the end of the string
useIndex	(default: TRUE) Print duplicate characters multiple times, as they appeared in the original matrix
byTaxon	If TRUE, write one taxon followed by the next. If FALSE, write one character followed by the next.
concatenate	Logical specifying whether to concatenate all characters/taxa into a single string, or to return a separate string for each entry.

Author(s)

Martin R. Smith

PrepareDataProfile *Prepare data for Profile Parsimony*

Description

Prepare data for Profile Parsimony

Usage

```
PrepareDataProfile(dataset, precision = 1e+06, warn = TRUE)

PrepareDataIW(dataset)
```

Arguments

dataset	dataset of class phyDat
precision	number of random trees to generate when calculating Profile curves. With 22 tokens (taxa): - Increasing precision from 4e+05 to 4e+06 reduces error by a mean of 0.005 bits for each step after the first (max = 0.11 bits, sd=0.017 bits) - Increasing precision from 1e+06 to 4e+06 reduces error by a mean of 0.0003 bits for each step after the first (max = 0.046 bits, sd=0.01 bits)
warn	Boolean (TRUE/FALSE): display warnings when concavity functions are generated by approximation.

Value

An object of class phyDat with additional attributes: `info.amounts`: details the information represented by each character when subject to N additional steps. `split.sizes`: The size of the splits implied by each character `bootstrap`: The character vector `c('info.amounts', 'split.sizes')`, indicating attributes to sample when bootstrapping the dataset 9e.g. in Ratchet searches).

Functions

- `PrepareDataIW`: Prepare data for implied weighting

Author(s)

Martin R. Smith; written with reference to phangorn:::prepareDataFitch

---

RandomMorphyTree *Random postorder tree*

Description

Random postorder tree

Usage

```
RandomMorphyTree(nTip)
```

Arguments

nTip	number of tips (minimum 3)

Value

A list with three elements, each a vector of integers, respectively containing: - The parent of each tip and node, in order - The left child of each node - The right child of each node.

RandomTree *Generate random tree topology from dataset*

Description

Generate random tree topology from dataset

Usage

```
RandomTree(dataset, root = FALSE)
```

Arguments

dataset	A dataset in phyDat format
root	Taxon to use as root (if desired; FALSE otherwise)

Author(s)

Martin R. Smith

RandomTreeScore *Parsimony score of random postorder tree*

Description

Parsimony score of random postorder tree

Usage

```
RandomTreeScore(nTip, morphyObj)
```

Arguments

nTip	number of tips (minimum 3)
morphyObj	A morphy object, perhaps created with PhyDat2Morphy.

Value

the parsimony score of a random tree, for the given Morphy object.

`ReadCharacters` *Read characters from Nexus file*

Description

Parses Nexus file, reading character states and names

Usage

```
ReadCharacters(filepath, character_num = NULL, session = NULL)

ReadTntCharacters(filepath, character_num = NULL, session = NULL)

ReadAsPhyDat(filepath)

ReadTntAsPhyDat(filepath)

PhyDat(dataset)
```

Arguments

`filepath`	character string specifying location of file
`character_num`	Index of character(s) to return. `NULL`, the default, returns all characters.
`session`	(optionally) a Shiny session with a numericInput named character_num whose maximum should be updated.
`dataset`	list of taxa and characters, in the format produced by read.nexus.data: a list of sequences each made of a single vector of mode character, and named with the taxon name.

Details

Tested with nexus files downloaded from MorphoBank with the "no notes" option, but should also work more generally.

Do report incorrectly parsed files.

Value

A matrix whose row names correspond to tip labels, and column names correspond to character labels, with the attribute `state.labels` listing the state labels for each character; or a character string explaining why the character cannot be returned.

Functions

- `ReadTntCharacters`: Read characters from TNT file
- `ReadAsPhyDat`: Read nexus characters as phyDat object
- `ReadTntAsPhyDat`: Read TNT characters as phyDat object

- PhyDat: A convenient wrapper for **phangorn**'s phyDat, which converts a *list* of morphological characters into a phyDat object. If your morphological characters are in the form of a *matrix*, perhaps because they have been read using read.table, try MatrixToPhyDat instead.

Author(s)

Martin R. Smith

Martin R. Smith

Martin R. Smith

References

Maddison, D. R., Swofford, D. L. and Maddison, W. P. (1997) NEXUS: an extensible file format for systematic information. Systematic Biology, 46, 590-621.

ReadTntTree *Parse TNT Tree*

Description

Reads a tree from TNT's paranthetical output.

Usage

```
ReadTntTree(filename, relativePath = NULL, keepEnd = 1L,
  tipLabels = NULL)

TNTText2Tree(treeText)
```

Arguments

filename	character string specifying path to TNT .tre file.
relativePath	(optional) character string specifying location of the matrix file used to generate the TNT results, relative to the current working directory, for portability. Taxon names will be read from this file if they are not specified by tipLabels.
keepEnd	(optional, default 1) integer specifying how many elements of the file path to conserve when creating relative path (see examples).
tipLabels	(optional) character vector specifying the names of the taxa, in the sequence that they appear in the TNT file. If not specified, taxon names will be loaded from the data file linked in the first line of the .tre file specified in filename.
treeText	Character string describing a tree, in the parenthetical format output by TNT.

Value

a tree of class phylo.

Functions

- `TNTText2Tree`: Converts text representation of a tree in TNT to an object of class `phylo`

Author(s)

Martin R. Smith

Martin R. Smith

Examples

```
{
  ## Not run:
  # TNT read a matrix from c:/myproject/tnt/coding1/dataset.nex
  # The results of an analysis were written to c:/myproject/tnt/output/results1.tnt
  # results1.tnt will contain a hard-coded reference to
  # "c:/myproject/tnt/coding1/dataset.nex"

  getwd() # Gives the current working directory

  # Say that working directory is c:/myproject, which perhaps corresponds to a
  # Git repository.
  # This directory may be saved into another location by collaborators, or on a
  # different filesystem by a continuous integration platform.

  # Works on local machine but not elsewhere:
  ReadTntTree('tnt/output/results1.tnt')

  # Takes only the filename from the results
  ReadTntTree('tnt/output.results1.tnt', 'tnt/coding1')

  # Uses the last three elements of c:/myproject/tnt/coding1/dataset.nex
  #                                              3     2       1
  # '.' means "relative to the current directory", which is c:/myproject
  ReadTntTree('tnt/output/results1.tnt', '.', 3)

  # If the current working directory was c:/myproject/rscripts/testing,
  # you could navigate up the directory path with '..':
  ReadTntTree('../../tnt/output/results1.tnt', '../..', 3)


## End(Not run)
}
```

`RearrangeEdges` *Rearrange edges of a phylogenetic tree*

Description

Rearranges a matrix that corresponds to the edges of a phylogenetic tree, returning the score of the new tree. Will generally be called from within a tree search function.

Usage

```
RearrangeEdges(parent, child, dataset, TreeScorer = MorphyLength,
  EdgeSwapper, scoreToBeat = TreeScorer(parent, child, dataset, ...),
  iter = "?", hits = 0L, verbosity = 0L, ...)
```

Arguments

`parent`	the first column of the edge matrix of a tree of class `phylo`, i.e. `tree$edge[, 1]`
`child`	the second column of the edge matrix of a tree of class `phylo`, i.e. `tree$edge[, 2]`
`dataset`	Third argument to pass to `TreeScorer`.
`TreeScorer`	function to score a given tree. The function will be passed three parameters, corresponding to the `parent` and `child` entries of a tree's edge list, and a dataset.
`EdgeSwapper`	a function that rearranges a parent and child vector, and returns a list with modified vectors; for example `SPRSwap`.
`scoreToBeat`	Double giving score of input tree.
`iter`	iteration number of calling function, for reporting to user only.
`hits`	Integer giving number of times the input tree has already been hit.
`verbosity`	Level of detail to display in console: larger numbers provide more verbose feedback to the user.
`...`	further arguments to pass to `TreeScorer` function (e.g. `TipsAreColumns`, `dataset`)

Details

`RearrangeTree` performs one tree rearrangement of a specified type, and returns the score of the tree (with the given dataset). It also reports the number of times that this score was hit in the current function call.

Value

This function returns a list with two to four elements, corresponding to a binary tree: - 1. Integer vector listing the parent node of each edge; - 2. Integer vector listing the child node of each edge; - 3. Score of the tree; - 4. Number of times that score has been hit.

Author(s)

Martin R. Smith

Examples

```
data('Lobo')
random.tree <- RandomTree(Lobo.phy)
edge <- random.tree$edge
parent <- edge[, 1]
child <- edge[, 2]
dataset <- PhyDat2Morphy(Lobo.phy)
RearrangeEdges(parent, child, dataset, EdgeSwapper=RootedNNISwap)
```

---

referenceTree *Tree topology for matrix simulation*

Description

The tree topology used to generate the matrices in congreveLamsdellMatrices Congreve & Lamsdell (2016)

Usage

```
referenceTree
```

Format

A single phylogenetic tree saved as an object of class phylo

Source

https://dx.doi.org/10.1111/pala.12236

References

Congreve CR, Lamsdell JC (2016). "Implied weighting and its utility in palaeontological datasets: a study using modelled phylogenetic matrices." *Palaeontology*, **59**, 447-465. doi: 10.1111/pala.12236.
Congreve CR, Lamsdell JC (2016). "Data from: Implied weighting and its utility in palaeontological datasets: a study using modelled phylogenetic matrices." *Dryad Digital Repository*, doi:10.5061/dryad.7dq0j. doi: 10.5061/dryad.7dq0j.

Examples

```
data(referenceTree)
plot(referenceTree)
```

Renumber *Renumber a tree's nodes and tips*

Description

Renumber numbers the nodes and tips in a tree to conform with the phylo standards.

Usage

```
Renumber(tree)
```

Arguments

tree	A tree of class phylo.

Value

This function returns a tree of class phylo

Author(s)

Martin R. Smith

Examples

```
library('ape')
tree <- rtree(10)
Renumber (tree)
```

RenumberTips *Reorder tips*

Description

RenumberTips(tree, tipOrder) sorts the tips of a phylogenetic tree such that the indices in tree$edge[, 2] correspond to the order of tips given in tipOrder

Usage

```
RenumberTips(tree, tipOrder)
```

Arguments

tree	A tree of class phylo.
tipOrder	A character vector containing the values of tree$tip.label in the desired sort order

Author(s)

Martin R. Smith

Examples

```
data(Lobo) # Loads the phyDat object Lobo.phy
tree <- RandomTree(Lobo.phy)
tree <- RenumberTips(tree, names(Lobo.phy))
```

RootTree *Root Tree on specified tips*

Description

Roots a tree on the smallest clade containing the specified tips.

Usage

```
RootTree(tree, outgroupTips)
```

Arguments

tree	A tree of class phylo.
outgroupTips	Character vector specifying the names of the tips to include in the outgroup.

Value

A tree of class phylo, rooted on the smallest clade that contains the specified tips

Author(s)

Martin R. Smith

SetMorphyWeights *Set the character weightings associated with a Morphy object*

Description

Set the character weightings associated with a Morphy object

Usage

```
SetMorphyWeights(weight, morphyObj, checkInput = TRUE)
```

Arguments

weight	A vector listing the new weights to be applied to each character
morphyObj	A morphy object, perhaps created with PhyDat2Morphy.
checkInput	Whether to sanity-check input data before applying. Defaults to TRUE to protect the user from crashes.

Value

The Morphy error code generated when applying tipData

Author(s)

Martin R. Smith

SingleCharMorphy *Morphy object from single character*

Description

Morphy object from single character

Usage

```
SingleCharMorphy(char)
```

Arguments

char	State of each character at each tip in turn, in a format that will be converted to a character string by paste0(char, ';', collapse='').

Value

A pointer to a morphyObj. Don't forget to unload it when you've finished with it: morphyObj <- UnloadMorphy(morphyObj)

Author(s)

Martin R. Smith

SingleTaxonTree *SingleTaxonTree*

Description

Single taxon tree

Usage

```
SingleTaxonTree(label)
```

Arguments

label	a character vector specifying the label of the tip.

Details

Create a phylogenetic 'tree' that comprises a single taxon.

Value

This function returns a phylo object containing a single tip with the specified label.

Examples

```
SingleTaxonTree('Homo_sapiens')
```

SortTree *Sort tree*

Description

Sorts each node into a consistent order, so similar trees look visually similar.

Usage

```
SortTree(tree)
```

Arguments

tree	A tree of class phylo.

Value

A tree of class phylo, with each node sorted such that the larger clade is first.

Author(s)

Martin R. Smith

SplitEntropy *Entropy of two splits*

Description

Reports various values pertaining to the phylogenetic information content of two splits, treating splits as subdivisions of *n* terminals into two clusters.

Usage

```
SplitEntropy(split1, split2 = split1)
```

Arguments

split1, split2	Logical vectors listing terminals in same order, such that each terminal is identified as a member of the ingroup (TRUE) or outgroup (FALSE) of the respective bipartition split.

Value

A numeric vector listing, in bits,

- h1 The entropy of split 1
- h1 The entropy of split 2
- jointH The joint entropy of both splits
- i The mutual information of the splits
- vI The variation of information of the splits (see Meila 2007)

Author(s)

Martin R. Smith

References

MeilÄf M (2007). "Comparing clusterings—an information based distance." *Journal of Multivariate Analysis*, **98**(5), 873–895. doi: 10.1016/j.jmva.2006.11.013.

SplitFrequency *Frequency of splits*

Description

SplitFrequency provides a simple way to count the number of times that bipartition splits, as defined by a reference tree, occur in a forest of trees.

Usage

```
SplitFrequency(reference, forest)

SplitNumber(tips, tree, tipIndex, powersOf2)

ForestSplits(forest, powersOf2)

TreeSplits(tree)
```

Arguments

reference	A tree of class phylo, or a character vector specifying its splits (as obtained through Tree2Splits)
forest	a list of trees of class phylo, or a multiPhylo object; or a list of their constituent splits (as obtained through Tree2Splits)
tips	Integer vector specifying the tips of the tree within the chosen split
tree	A tree of class phylo.
tipIndex	Character vector of tip names, in a fixed order
powersOf2	Integer vector of same length as tipIndex, specifying a power of 2 to be associated with each tip in turn

Details

If multiple calculations are required, some time can be saved by using the constituent functions (see examples)

Value

Number of trees in forest that contain each split in reference. if reference is a tree of class phylo, then the sequence will correspond to the order of nodes (use 'ape::nodelabels to view). Note that the three nodes at the root of the tree correspond to a single split; see the example for how these might be plotted on a tree.

Functions

- SplitNumber: Assign a unique integer to each split
- ForestSplits: Frequency of splits in a given forest of trees
- TreeSplits: Deprecated. Listed the splits in a given tree. Use Quartet::Tree2Splits instead.

Author(s)

Martin R. Smith

Examples

```
{
  library(ape) # for functions rtree & nodelabels
  set.seed(0) # Set seed so random trees are reproducable
  tree1 <- rtree(7)
  tree2 <- rtree(7)
  tree3 <- rtree(7)
  forest <- list(tree1, tree2, tree2, tree3, rtree(7))

  # Simple, but means counting each split in the forest twice:
  tree1Freqs <- SplitFrequency(tree1, forest)
  SplitFrequency(tree2, forest)

  plot(tree1)
  nodelabels(tree1Freqs, node=as.integer(names(tree1Freqs)))
}
```

SplitInformation *Information content of a split*

Description

`SplitInformation` calculates the information content of a split, based on the entropy of the subset of trees consistent with the split; a split that is consistent with a smaller number of trees will have a higher information content.

Usage

```
SplitInformation(A, B)

MultiSplitInformation(partitionSizes)
```

Arguments

`A`	Number of taxa in each partition.
`B`	Number of taxa in each partition.
`partitionSizes`	Integer vector specifying the number of taxa in each partition of a multi-partition split.

Value

Information content of the split, in bits.

Functions

- `MultiSplitInformation`: Information content of a multi-partition split.

Author(s)

Martin R. Smith

Examples

```
# Eight tips can be split evenly:
SplitInformation (4, 4)

# or unevenly, which is less informative:
SplitInformation (2, 6)
```

SplitMatchProbability *Probability of matching this well*

Description

Calculates the probability that two random splits of the sizes provided will be at least as similar as the two specified.

Usage

```
SplitMatchProbability(split1, split2)

LnSplitMatchProbability(split1, split2)
```

Arguments

`split1, split2`	Logical vectors listing terminals in same order, such that each terminal is identified as a member of the ingroup (`TRUE`) or outgroup (`FALSE`) of the respective bipartition split.

Value

The proportion of permissable informative splits splitting the terminals into bipartitions of the sizes given, that match as well as `split1` and `split2` do.

Functions

- `LnSplitMatchProbability`: The natural logarithm of the probability

Author(s)

Martin R. Smith

Examples

```
SplitMatchProbability(split1 = c(rep(TRUE, 4), rep(FALSE, 4)),
                      split2 = c(rep(TRUE, 3), rep(FALSE, 5)))
```

SplitMutualInformation

Mutual information of two splits

Description

Reports the mutual phylogenetic information, or variation of phylogenetic information, of two splits.

Usage

```
SplitMutualInformation(n, A1, A2 = A1)

SplitVariationOfInformation(n, A1, A2 = A1)

TreesConsistentWithTwoSplits(n, A1, A2 = A1)

LogTreesConsistentWithTwoSplits(n, A1, A2 = A1)
```

Arguments

`n`	Integer specifying the number of terminals.
`A1, A2`	Integers specifying the number of taxa in *A1* and *A2*, once the splits have been arranged such that *A1* overlaps with *A2*.

Details

The mutual phylogenetic information corresponds to the entropy of the subset of trees consistent with both splits; two splits that are consistent with a smaller number of trees will have a higher mutual information content.

Split 1 divides `n` terminals into two partitions, *A1* and *B1*. Split 2 divides the same terminals into the partitions *A2* and *B2*.

Partitions must be named such that *A1* overlaps with *A2*: that is to say, all taxa in *A1* are also in *A2*, or *vice versa*. Thus, all taxa in the smaller of *A1* and *A2* also occur in the larger.

Value

`TreesConsistentWithTwoSplits` returns the number of unrooted bifurcating trees consistent with two splits.

`SplitMutualInformation` returns the information that two splits have in common, in bits.

`SplitVariationOfInformation` returns the variation of information (Meila 2007) between the two splits, a measure of their difference, in bits.

Functions

- `SplitVariationOfInformation`: Variation of information between two splits.
- `TreesConsistentWithTwoSplits`: Number of trees consistent with two splits.
- `LogTreesConsistentWithTwoSplits`: Natural logarithm of `TreesConsistentWithTwoSplits`.

Author(s)

Martin R. Smith

References

MeilÄf M (2007). "Comparing clusterings—an information based distance." *Journal of Multivariate Analysis*, **98**(5), 873–895. doi: 10.1016/j.jmva.2006.11.013.

Examples

```
# Eight tips, labelled A to H.
# Split 1: ABCD:EFGH
# Split 2: ABC:DEFGH
# Let A1 = ABCD (four taxa), and A2 = ABC (three taxa).
# A1 and A2 overlap (both contain ABC).

TreesConsistentWithTwoSplits(n=8, A1=4, A2=3)
SplitMutualInformation(n=8, A1=4, A2=3)
SplitVariationOfInformation(n=8, A1=4, A2=3)

# If splits are identical, then their mutual information is the same
# as the information of either split:
SplitMutualInformation(n=8, A1=3, A2=3)
SplitInformation(3, 5)
```

SplitsCompatible *Are splits compatible?*

Description

Splits are compatible if they are concave; i.e. they can both be true simultaneously.

Usage

```
SplitsCompatible(split1, split2)
```

Arguments

`split1, split2`	Logical vectors listing terminals in same order, such that each terminal is identified as a member of the ingroup (`TRUE`) or outgroup (`FALSE`) of the respective bipartition split.

Value

SplitsCompatible returns a logical specifying whether the splits provided are compatible with one another.

Author(s)

Martin R. Smith

SPR *Subtree Pruning and Rearrangement (SPR)*

Description

Perform one SPR rearrangement on a tree

Usage

```
SPR(tree, edgeToBreak = NULL, mergeEdge = NULL)

SPRSwap(parent, child, nEdge = length(parent), nNode = nEdge/2L,
  edgeToBreak = NULL, mergeEdge = NULL)

RootedSPR(tree, edgeToBreak = NULL, mergeEdge = NULL)

RootedSPRSwap(parent, child, nEdge = length(parent), nNode = nEdge/2L,
  edgeToBreak = NULL, mergeEdge = NULL)
```

Arguments

tree	A tree of class phylo.
edgeToBreak	the index of an edge to bisect, generated randomly if not specified.
mergeEdge	the index of an edge on which to merge the broken edge.
parent	the first column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 1]
child	the second column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 2]
nEdge	(optional) integer specifying the number of edges of a tree of class phylo, i.e. dim(tree$edge)[1]
nNode	(optional) Number of nodes.

Details

Equivalent to phangorn's kSPR, but faster. Note that rearrangements that only change the position of the root WILL be returned by SPR. If the position of the root is irrelevant (as in Fitch parsimony, for example) then this function will occasionally return a functionally equivalent topology. RootIrrelevantSPR will search tree space more efficiently in these cases. Branch lengths are not (yet) supported.

All nodes in a tree must be bifurcating; ape::collapse.singles and ape::multi2di may help.

Value

This function returns a tree in phyDat format that has undergone one SPR iteration.

a list containing two elements, corresponding in turn to the rearranged parent and child parameters

a list containing two elements, corresponding in turn to the rearranged parent and child parameters

Functions

- SPRSwap: faster version that takes and returns parent and child parameters
- RootedSPR: Perform SPR rearrangement, retaining position of root
- RootedSPRSwap: faster version that takes and returns parent and child parameters

Author(s)

Martin R. Smith

References

The SPR algorithm is summarized in Felsenstein J (2004). *Inferring phylogenies*. Sinauer Associates, Sunderland, Massachusetts.

See Also

RootedSPR useful when the position of the root node should be retained.

TBR

NNI

Examples

```
{
tree <- ape::rtree(20, br=FALSE)
SPR(tree)
}
```

StringToPhyDat *String to phyDat*

Description

Converts a PhyDat object to allow processing by MorphyDat

Usage

```
StringToPhyDat(string, tips, byTaxon = TRUE)
```

Arguments

string	a string of tokens, optionally containing whitespace, with no terminating semi-colon. Polytomies not (yet) supported; each character must correspond to a unique state, ?, or the inapplicable token (-)
tips,	a character vector corresponding to the names (in order) of each taxon in the matrix
byTaxon	= TRUE, string is one TAXON's coding at a time; FALSE: one CHARACTER's coding at a time

Value

This function returns a data matrix in phyDat format.

Author(s)

Martin R. Smith

See Also

phyDat

Examples

```
morphy <- StringToPhyDat("-?01231230?-", c('Lion', 'Gazelle'), byTaxon=TRUE)
# encodes the following matrix:
# Lion     -?0123
# Gazelle  1230?-
```

Subtree *Extract subtree*

Description

Safely extracts a clade from a phylogenetic tree.

Usage

```
Subtree(tree, node)
```

Arguments

tree	A tree of class phylo, with internal numbering in cladewise order (use Preorder(tree) or (slower) Cladewise(tree))
node	The number of the node at the base of the clade to be extracted.

Details

Modified from the **ape** function `extract.clade`, which sometimes behaves erratically. Unlike extract.clade, this function supports the extraction of 'clades' that constitute a single tip.

Value

This function returns a tree of class `phylo` that represents a clade extracted from the original tree.

Author(s)

Martin R. Smith

Examples

```
{
tree <- Preorder(ape::rtree(20, br=NULL))
plot(tree); ape::nodelabels(); ape::nodelabels(33, 33, bg='yellow'); dev.new()
plot(Subtree(tree, 33))
}
```

SuccessiveApproximations

Tree Search using Successive Approximations

Description

Searches for a tree that is optimal under the Successive Approximations criterion

Usage

```
SuccessiveApproximations(tree, dataset, outgroup = NULL, k = 3,
  maxSuccIter = 20, ratchetHits = 100, searchHits = 50,
  searchIter = 500, ratchetIter = 5000, verbosity = 0,
  suboptimal = 0.1)
```

Arguments

`tree`	A tree of class `phylo`.
`dataset`	A phylogenetic data matrix of class `phyDat`, whose names correspond to the labels of any accompanying tree.
`outgroup`	if not NULL, taxa on which the tree should be rooted
`k`	Constant for successive approximations, see Farris 1969 p. 379
`maxSuccIter`	maximum iterations of successive approximation
`ratchetHits`	maximum hits for parsimony ratchet
`searchHits`	maximum hits in tree search

searchIter	maximum iterations in tree search
ratchetIter	maximum iterations of parsimony ratchet
verbosity	integer (default 0) specifying how much detail to print to stdout
suboptimal	retain trees that are this proportion less optimal than the optimal tree

Value

list of optimal (and slightly suboptimal, if suboptimal > 0) trees

---

summary.morphyPtr *Details the attributes of a morphy object*

Description

Details the attributes of a morphy object

Usage

```
## S3 method for class 'morphyPtr'
summary(object, ...)
```

Arguments

object	A Morphy object
...	any other parameters...

Value

A list detailing the number of taxa, internal nodes, and characters and their weights.

Author(s)

Martin R. Smith

SupportColour *Support colour*

Description

Support colour

Usage

```
SupportColour(support, show1 = TRUE)

SupportColor(support, show1 = TRUE)
```

Arguments

support	A vector of doubles in the range 0-1
show1	Logical specifying whether to display values of 1. A transparent white will be returned if FALSE.

Value

A string containing the hexadecimal code for a colour picked from a diverging scale, or red if a value is invalid.

Functions

- SupportColor: alternative spelling

TBR *Tree bisection and reconnection (TBR)*

Description

TBR performs a single random TBR iteration.

Usage

```
TBR(tree, edgeToBreak = NULL, mergeEdges = NULL)

TBRSwap(parent, child, nEdge = length(parent), edgeToBreak = NULL,
  mergeEdges = NULL)

TBRMoves(parent, child, nEdge = length(parent), avoid = NULL,
  retainRoot = FALSE)
```

```
AllTBR(parent, child, nEdge = length(parent), avoid = NULL,
  retainRoot = FALSE)

RootedTBR(tree, edgeToBreak = NULL, mergeEdges = NULL)

RootedTBRSwap(parent, child, nEdge = length(parent),
  edgeToBreak = NULL, mergeEdges = NULL)
```

Arguments

tree	A bifurcating tree of class phylo, with all nodes resolved;
edgeToBreak	(optional) integer specifying the index of an edge to bisect/prune, generated randomly if not specified. Alternatively, set to -1 to return a complete list of all trees one step from the input tree.
mergeEdges	(optional) vector of length 1 or 2, listing edge(s) to be joined: In SPR, this is where the pruned subtree will be reconnected. In TBR, these edges will be reconnected (so must be on opposite sides of edgeToBreak); if only a single edge is specified, the second will be chosen at random
parent	the first column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 1]
child	the second column of the edge matrix of a tree of class phylo, i.e. tree$edge[, 2]
nEdge	(optional) Number of edges.
avoid	Integer vector specifying which edges should not be broken
retainRoot	logical specifying whether taxa may be swapped across the root

Details

Branch lengths are not (yet) supported.

All nodes in a tree must be bifurcating; ape::collapse.singles and ape::multi2di may help.

Value

This function returns a tree in phyDat format that has undergone one TBR iteration.

a list containing two elements, corresponding in turn to the rearranged parent and child parameters

a matrix with two columns, each row listing an edge that can be broken and an edge into which it can be merged

a list of trees, in parent-child format

Functions

- TBRSwap: faster version that takes and returns parent and child parameters
- TBRMoves: Possible TBR moves
- AllTBR: All unique trees one TBR move away
- RootedTBR: Perform TBR rearrangement, retaining position of root
- RootedTBRSwap: faster version that takes and returns parent and child parameters

Author(s)

Martin R. Smith

References

The TBR algorithm is summarized in Felsenstein J (2004). *Inferring phylogenies*. Sinauer Associates, Sunderland, Massachusetts.

See Also

RootedTBR useful when the position of the root node should be retained.

Examples

```
{
library('ape')
tree <- rtree(20, br=NULL)
TBR(tree)
}
```

TreesMatchingSplit *Number of trees matching a bipartition split*

Description

Calculates the number of unrooted bifurcated trees that are consistent with a bipartition split that divides taxa into groups of size A and B.

Usage

```
TreesMatchingSplit(A, B)

LogTreesMatchingSplit(A, B)
```

Arguments

A, B	Number of taxa in each partition.

Functions

- LogTreesMatchingSplit: Logarithm of the number of trees matching a split.

Author(s)

Martin R. Smith

UniqueSplits *Unique Splits*

Description

Removes equivalent duplicates from a matrix of bipartitions.

Usage

```
UniqueSplits(splits, preserveParity = FALSE)
```

Arguments

splits	A logical matrix containing one named row corresponding to each terminal leaf of a tree, and each column corresponds to a bipartition split; each split divides terminals into two bipartitions; members of one are marked TRUE and members of the other are marked FALSE.
preserveParity	Logical specifying whether to preserve the TRUE and FALSE status within each split (which takes marginally longer). If FALSE, each split will be defined such that taxa in the same partition as the first element are marked FALSE, and other taxa marked TRUE.

Value

The splits element, with all duplicate splits removed.

Author(s)

Martin R. Smith

Examples

```
set.seed(1)
splits <- Tree2Splits(ape::rtree(6, br=NULL))
UniqueSplits(splits, preserveParity=TRUE)
```

UnloadMorphy *Destroy a Morphy Object*

Description

Best practice is to call `morphyObj <- UnloadMorphy(morphyObj)` Failure to do so will cause a crash if UnloadMorphy is called on an object that has already been destroyed

Usage

```
UnloadMorphy(morphyObj)
```

Arguments

morphyObj	A morphy object, perhaps created with PhyDat2Morphy.

Value

Morphy error code, decipherable using mpl_translate_error

Author(s)

Martin R. Smith

UnloadTreeSearch *Unload this library*

Description

Unload this library

Usage

```
UnloadTreeSearch()
```

UnrootedTreesMatchingSplit *Number of trees consistent with split*

Description

Calculates the number of unrooted bifurcating trees consistent with the specified multi-partition split, using the formula of Carter *et al*. (1990).

Usage

```
UnrootedTreesMatchingSplit(splits)
```

Arguments

splits	A vector of integers listing the number of tips in each of a number of tree splits (e.g. bipartitions). For example, `c(3, 5)` states that a character divides a set of eight tips into a group of three and a group of five.

Value

UnrootedTreesMatchingSplit returns an integer specifying the number of unrooted bifurcating trees consistent with the specified split.

Author(s)

Martin R. Smith

References

Carter M, Hendy M, Penny D, SzÃ©kely LA, Wormald NC (1990). "On the distribution of lengths of evolutionary trees." *SIAM Journal on Discrete Mathematics*, **3**(1), 38–47. doi: 10.1137/0403005, http://epubs.siam.org/doi/abs/10.1137/0403005., Theorem 2.

Examples

```
UnrootedTreesMatchingSplit(c(3, 5))
UnrootedTreesMatchingSplit(c(3, 2, 1, 2))
```

VisualizeMatching *Visualise a matching*

Description

Depicts the bipartitions that are matched between two trees using a specified Generalized Robinson Foulds tree distance measure.

Usage

```
VisualizeMatching(Func, tree1, tree2, setPar = TRUE, Plot = plot.phylo,
  ...)
```

Arguments

Func	Function used to construct tree similarity.
tree1, tree2	Trees of class phylo, with tips labelled identically.
setPar	Logical specifying whether graphical parameters should be set to display trees side by side.
Plot	Function to use to plot trees.
...	Additional parameters to send to Plot.

Author(s)

Martin R. Smith

`WithOneExtraStep` *Number of trees with one extra step*

Description

Number of trees with one extra step

Usage

```
WithOneExtraStep(splits)
```

Arguments

`splits`	A vector of integers listing the number of tips in each of a number of tree splits (e.g. bipartitions). For example, `c(3, 5)` states that a character divides a set of eight tips into a group of three and a group of five.

Index