Package ‘megaptera’

February 20, 2015

Type Package
Title MEGAPhylogeny Techniques in R
Version 1.0-0
Date 2014-10-30
Author Christoph Heibl
Maintainer Christoph Heibl <christoph.heibl@gmx.net>
Depends R (>= 2.10), snowfall, ips
Imports ape, seqinr, methods, RPostgreSQL, XML
Suggests knitr
Description Given a set of taxa and marker, the goal is build a complete and correct sequence alignment for phylogenetic analysis.
License GPL (>= 2)
VignetteBuilder knitr
NeedsCompilation no
Repository CRAN
Date/Publication 2014-11-13 08:44:56

R topics documented:

megaptera-package .............................................. 2
addTip ......................................................... 3
cetacea ....................................................... 4
check.Functions ............................................. 5
dbPars .......................................................... 6
dbPars-class .................................................. 6
dbReadTaxonomy ............................................. 7
dbUpdateTaxonomy .......................................... 8
fleshoutGuidetree ........................................... 8
io.dna.db ..................................................... 9
locus .......................................................... 10
locus-class ............................................... 11
Description

The goal of developing **megaptera** is the automation of sequence-retrieval from internet databases (currently Nucleotide at GenBank) to produce (possibly large) aligned nucleotide sequence datasets of one or more genetic markers for **comparative analyses**.

Details

Package:    megaptera
Type:       Package
Version:    1.0-0
Date:       2014-10-30
License:    GPL (>= 2)

Author(s)

Christoph Heibl Maintainer: Christoph Heibl <christoph.heibl@gmx.net>
addTip

References

See Also
ape

addTip Function
Add Tips to a Phylogeny

Description
This function adds tips (terminal nodes) to a (currently ultrametric) phylogenetic tree.

Usage
addTip(phy, tip, tax, insert = "crown", stem.edge = 0.5)

Arguments
- phy: an object of class phylo; must be ultrametric.
- tip: a character string giving the name of the species to be added.
- tax: a data frame containing a taxonomy.
- insert: a character string indicating the positions where the species is to be inserted: "crown", "stem", "randomly", or any unambiguous abbreviation of these. This option will only have an effect if phy contains more than one congeneric of tip.
- stem.edge: a real number greater than 0 and smaller than 1, which gives the fraction of the terminal branch length that will be assigned to the branch subtending the newly created MRCA of tip and its single congeneric species in phy. Will have no effect if phy contains more than one congeneric of tip.

Details
tip will be added at the crown group level of the genus. This means there must be a monophyletic generic clade in the phylogeny that matches the genus name of tip.

Value
an object of class phylo containing at least one multichotomous node.

Author(s)
Christoph Heibl

See Also
prune.phylo.rank for pruning phylogenies to a particular taxonomic level.
Examples

```r
# random ultrametric tree
# -------------------
#set.seed(12)
#phy <- rcoal(20, tip.label = paste("whale", LETTERS[1:20]))
#phy$tip.label[12] <- gsub("whale", "humpback", phy$tip.label[12])
#phy <- fixNodes(ladderize(phy))
#plot(phy, no.margin = TRUE); nodelabels(cex = .5)

# default use adding four species
# -------------------------------
#spec <- c("Athelia V", 
#          paste("Cladonia", c("X", "Y", "Z")))
#phy.crown <- phy
#for (i in seq_along(spec))
#  phy.crown <- addTip(phy.crown, spec[i])
#plot(phy.crown, no.margin = TRUE); nodelabels(cex = .5)

# random addition of tips to their genus
# -------------------------------------
#spec <- paste("Cladonia", c("X", "Y", "Z"))
#phy.random <- phy
#for (i in seq_along(spec))
#  phy.random <- addTip(phy.random, spec[i], insert = "randomly")
#plot(phy.random, no.margin = TRUE); nodelabels(cex = .5)
```

cetacea

**Cetacea Phylogeny**

**Description**

Phylogenetic tree of whales (Cetacea).

**Usage**

data("cetacea")

**Format**

The format is:

`List of 5 $ edge : num [1:178, 1:2] 91 92 93 94 95 96 97 98 99 100 ... $ Nnode : int 89 $ tip.label : chr [1:90] "Delphinus_capidensis" "Delphinus_delphis" "Tursiops_aduncus" "Stenella_frontalis" ... $ edge.length: num [1:178] 0.05338 0.00533 0.00864 0.0074 0.02409 ... $ node.label : chr [1:89] "100" "35" "57" ... - attr(*, "class")= chr "phylo" - attr(*, "order")= chr "cladewise"

**Source**

The phylogeny is based on a supermatrix of five loci assembled with `megaptera` on November 6, 2014 and was estimated under a GTR + Gamma model of sequence evolution in RAxML 8.0.20.
check.Functions

Examples

data(cetacea)

Description

This set of functions helps to check progress and consistency of a megaptera project.

Usage

check.NameChanges(conn, order.by = "spec")
check.Species(conn, spec, gene)
check.Markers(conn, subset, outgroup, colname = "blocks", plot = TRUE)
check.Coverage(conn, pool.markers = TRUE)
check.Clades(phy, tax)

Arguments

conn a database connection object.
spec character string giving the name of a species
gene character string giving the name of a genetic marker
subset character string giving the names of a subset of species
outgroup character string giving the name(s) of outgroup species
order.by a character string giving the column name that is used for ordering the output table
colname character string: gb, sel, or blocks.
plot logical
pool.markers logical
phy an object of class phylo
tax a data frame containing a taxonomic classification

Value

a data frame

Author(s)

Christoph Heibl

References

**dbPars-class**

---

### Description

Sets database parameters.

### Usage

```r
dbPars(host = "localhost", port = 5432, dbname, user = "postgres", password)
```

### Arguments

- **host**: a vector of mode "character"
- **port**: numeric
- **dbname**: a vector of mode "character"
- **user**: a vector of mode "character"
- **password**: a vector of mode "character"

### Value

An object of class `dbPars`.

### Author(s)

Christoph Heibl

### See Also

`taxon`, `locus`, and `megapteraPars` for defining of taxa, loci, and the pipeline's parameters, respectively, and `megapteraProj` for the bundling of input data.

---

### dbPars-class

**Class** "dbPars"

---

### Description

Class for database parameters for `megapteraProj`.

### Objects from the Class

Objects can be created by calls of the form `dbPars(host = "localhost", port = 5432, dbname, user = "postgres", password)`. 
Slots

- **host**: a vector of mode "character"
- **port**: numeric
- **dbname**: a vector of mode "character"
- **user**: a vector of mode "character"
- **password**: a vector of mode "character"

Methods

- **show** signature(object = "taxon"): prints taxonomic parameter setting

Author(s)

Christoph Heibl

See Also

dbPars, locus, and megapteraPars for defining of database parameters, loci, and the pipeline’s parameters, respectively, and megapteraProj for the bundling of input data.

dbReadTaxonomy  Taxonomy of Species Contained in SQL Database

Description

Queries the SQL database and return a data frame containing the taxonomy of all species currently contained in the database.

Usage

dbReadTaxonomy(conn, subset)

Arguments

- **conn**: a connection object as produced by dbConnect.
- **subset**: a subset of species names the taxonomy should be limited to. Can be a DNA alignment of class DNAbin, a data frame or a simple vector enumerating the species names.

Value

- an object of class "data.frame" containing the taxonomic classification of a set of target species.

Author(s)

Christoph Heibl
See Also

dbReadDNA, dbWriteDNA

dbUpdateTaxonomy       Taxonomy Table

Description

Creates or extends the taxonomy table of a pgSQL database.

Usage

dbUpdateTaxonomy(conn, taxonomy)

Arguments

conn             a connection object as produced by dbConnect
taxonomy         a data frame containing a taxonomic classification of the study species.

Value

None. dbUpdateTaxonomy is called for its side effect creating and extending the taxonomy database table.

Author(s)

Christoph Heibl

See Also

ncbiTaxonomy for retrieval of taxonomies from the taxonomy database at NCBI.

fleshoutGuidetree       Add Missing Species to a Phylogenetic Tree

Description

Species missing in a phylogenetic tree will be added to that tree according to their taxonomic classification.

Usage

fleshoutGuidetree(phy, tax)
io.dna.db

Arguments

phy an object of class phylo.
tax a data frame containing the study groups’ taxonomy.

Value

an object of class phylo.

Author(s)

Christoph Heibl

See Also
dbReadTaxonomy for reading a taxonomy table from the postgreSQL database; addTip for addition of single tips manually.

Description

DNA sequence alignment can be read from and written to a (already existing) pgSQL database table.

Usage

dbReadDNA(conn, tab.name, taxon, regex = FALSE, max.bp,
max.dist, enforce.binomial = FALSE, ignore.excluded =
TRUE)

dbWriteDNA(conn, tab.name, seqs, enforce.binomial = TRUE,
status)

Arguments

conn a connection object as produced by dbConnect.
tab.name a character string giving the name of the database table (gene).
taxon a character string giving the name of the taxon.
regex logical: if TRUE, the string given via taxon will be interpreted as a regular expression (see regex).
seqs an object of class DNAbin to write to the database.
enforce.binomial Logical.
max.bp an integer, only sequences shorter equal or shorter than max.bp will be returned.
max.dist an integer, only sequences equally or less distant to the benchmark than max.dist will be returned.

status a character string to be written to the status field of the PgSQG table.

ignore.excluded logical: if TRUE, dbReadDNA will return also sequences that are marked in the status field as 'excluded', 'too long', or 'too distant'.

Author(s)
Christoph Heibl

locus Locus/Phylogenetic Marker Definition

Description
Creates a S4 class defining a phylogenetic marker for a megapteraProj.

Usage
locus(..., not, search.fields = c("gene", "title"),
       check = TRUE)

Arguments
... a vector of mode character giving strings that should be searched for; the first element is taken to name the corresponding PostgreSQL tables.
not a vector of mode character giving strings that should be excluded from the search results; corresponds to the use of NOT in GenBank query.
search.fields a vector of mode character setting the search fields (or attributes) of the Nucleotide database to be searched for the strings specified via the ... argument.
check logical: if TRUE, the existence of a locus as specified by ... and not will be checked.

Value
an object of class locus

Author(s)
Christoph Heibl

See Also
dbPars, taxon, and megapteraPars for defining of database parameters, taxa, and the pipeline’s parameters, respectively, and megapteraProj for the bundling of input data.
**locus-class**

Class "locus"

**Description**

Class for genetic input parameters for megapleraProj.

**Objects from the Class**

Objects can be created by calls of the form locus(..., not, search.fields = c("gene", "title").

**Slots**

- **sql** a vector of mode character holding a SQL-compatible transformation of the first alias for naming database tables.
- **aliases** a vector of mode character giving strings that should be searched for; the first element is taken to name the corresponding postgresQL tables.
- **not** a vector of mode character giving strings that should be excluded from the search results; corresponds to the use of NOT in GenBank query.
- **search.fields** a vector of mode character setting the search fields (or attributes) of the Nucleotide database to be searched for the strings specified via the ... argument.

**Author(s)**

Christoph Heibl

**See Also**

dbpars, taxon, and megapleraPars for defining of database parameters, taxa, and the pipeline’s parameters, respectively, and megapleraProj for the bundling of input data.

---

**megapleraPars**  

Set Pipeline Parameters

**Description**

Sets size limits, threshold values and other parameters that control the flow along the pipeline.

**Usage**

megapleraPars(...)

**Arguments**

... arguments in tag = value form. The tags must come from the names of graphical parameters described in the ‘Parameters’ section.
Value

a list of parameters described in the 'Parameters' section.

Parameters

update.seqs currently unused.

max.gi.per.spec numeric, giving the maximum number of sequences that will be used per species. Can be used to avoid model organism (e.g., rice, Drosophila, ...) cluttering up the pipeline with thousands of sequences (default: 100).

max.bp numeric, the maximal length of DNA sequences in base pairs to be included in the alignment. The upper limit is determined by the alignment program and the specific alignment and can only be determined by trial-and-error (default: 5000).

reference.stringence currently unused.

reference.max.dist currently unused.

min.seqs.reference currently unused.

max.dist numeric, ranging between 0 and 1. The sequences from taxa for which only a single accession is available will be included only if their genetic distance (measured as the fraction of divergent nucleotide positions) is less then max.dist (default: 0.5)

fract.miss numeric, ranging between 0 and 1. To avoid long stretches of only a few sequences at the beginning and the ending of an alignment block a minimum required number of sequences can be set as a fraction of the total number of sequences in this alignment block. Has been superseeded by the gb.* parameters.

filter1 currently unused.

filter2 currently unused.

filter3 currently unused.

filter4 currently unused.

block.max.dist numeric, ranging between 0 and 1. block.max.dist gives the maximum genetic distance (measured as the fraction of divergent nucleotide positions) allowed in a sequence alignment block. The alignment of individual marker is iteratively broken into smaller blocks until this condition is met with.

min.n.seq numeric, the minimum number of sequences required for an alignment block. Alignment blocks with less than min.n.seq are dropped from the output.

gb1, gb2, gb3, gb4, gb5 parameters for masking of alignment blocks with gblocks.

Author(s)

Christoph Heibl

See Also

megapteraproj for creating a megaptera project.

Examples

megapteraPars()
megapteraPars-class

megapteraPars-class  Class "megapteraPars"

Description

Class for pipeline parameters of megapteraProj.

Objects from the Class

Objects can be created by calls of the form megapteraPars(...).

Slots

- **update.seqs**  currently unused.
- **max.gi.per.spec**  numeric, giving the maximum number of sequences that will be used per species. Can be used to avoid model organism (e.g., rice, Drosophila, ...) cluttering up the pipeline with thousands of sequences (default: 100).
- **max.bp**  numeric, the maximal length of DNA sequences in base pairs to be included in the alignment. The upper limit is determined by the alignment program and the specific alignment and can only be determined by trial-and-error (default: 5000).
- **reference.stringence**  currently unused.
- **reference.max.dist**  currently unused.
- **min.seqs.reference**  currently unused.
- **max.dist**  numeric, ranging between 0 and 1. The sequences from taxa for which only a single accession is available will be included only if their genetic distance (measured as the fraction of divergent nucleotide positions) is less than max.dist (default: 0.5).
- **fract.miss**  numeric, ranging between 0 and 1. To avoid long stretches of only a few sequences at the beginning and the ending of an alignment block a minimum required number of sequences can be set as a fraction of the total number of sequences in this alignment block. Has been superseeded by the gb.* parameters.
- **filter1**  currently unused.
- **filter2**  currently unused.
- **filter3**  currently unused.
- **filter4**  currently unused.
- **block.max.dist**  numeric, ranging between 0 and 1. block.max.dist gives the maximum genetic distance (measured as the fraction of divergent nucleotide positions) allowed in a sequence alignment block. The alignment of individual marker is iteratively broken into smaller blocks until this condition is met with.
- **min.n.seq**  numeric, the minimum number of sequences required for an alignment block. Alignment blocks with less than min.n.seq are dropped from the output.
- **gb1, gb2, gb3, gb4, gb5**  parameters for masking of alignment blocks with gblocks.
megapertaProj

Author(s)

Christoph Heibl

See Also

megapertaProj for creating a megaperta project.

Examples

megapertaPars()

megapertaProj                  Bundle Input Data for the Pipeline

Description

Create an object of class megapertaProj to hold all the input data necessary for running of the pipeline.

Usage

megapertaProj(db, taxon, locus = locus(),
  align.exe, mask.exe,
  params = megapertaPars(),
  update = FALSE, parallel = FALSE)

Arguments

db       an object of class dbPars.
taxon    an object of class taxon.
locus    an object of class locus.
align.exe a vector of mode "character", giving name of the alignment program; currently only MAFFT is allowed
mask.exe a vector of mode "character", giving name of the alignment masking program; currently only Gblocks is allowed
params   an object of class megapertaPars.
update   logical: if TRUE, the pipeline’s steps are executed as if called fro the first time, i.e., possibly overriding data and setting that have been previously achieved.
parallel logical: if TRUE, several step in the pipeline will be run in parallel, otherwise all steps are serial.

Value

an object of class megapertaProj
megapteraProj-class

Author(s)
Christoph Heibl

See Also
dbPars, taxon, locus, and megapteraPars for defining of database parameters, taxa, loci, and the pipeline’s parameters, respectively.

megapteraProj-class  Class "megapteraProj"

Description
Class for holding all input data of the pipeline.

Objects from the Class
Objects can be created by calls of the form megapteraProj(db, taxon, locus, align.exe, mask.exe, params = megapteraPars)

Slots

  db  an object of class dbPars.
  taxon  an object of class taxon.
  locus  an object of class locus.
  align.exe  a vector of mode "character", giving name of the alignment program; currently only MAFFT is allowed
  mask.exe  a vector of mode "character", giving name of the alignment masking program; currently only Gblocks is allowed
  params  an object of class megapteraPars.
  update  logical: if TRUE, the pipeline’s steps are executed as if called fro the first time, i.e., possibly overriding data and setting that have been previously achieved.
  parallel  logical: if TRUE, several step in the pipeline will be run in parallel, otherwise all steps are serial.

Author(s)
Christoph Heibl

See Also
dbPars, taxon, locus, and megapteraPars for defining of database parameters, taxa, loci, and the pipeline’s parameters, respectively.
NCBIeaa

NCBI Extended Amino Acid

Description

This table provides a key to the Extended IUPAC Encoding of Amino Acids defined by NCBI.

Usage

data(NCBIeaa)

Format

A data frame with 28 observations on the following 4 variables:

- Value  NCBI code
- Symbol  One-letter abbreviation of amino acids
- Symbol3  Three-letter abbreviation of amino acids
- Name  Amino acids

Details

The official IUPAC amino acid code has some limitations. One is the lack of symbols for termination, gap, or selenocysteine. Such extensions to the IUPAC codes are also commonly used by sequence analysis software. NCBI has created such a code which is simply the IUPACaa code above extended with the additional symbols.

Source

http://www.ncbi.nlm.nih.gov/books/NBK7198/#ch_datamod.datamodel.seqloc

References

http://www.ncbi.nlm.nih.gov/books/NBK7198/#ch_datamod.datamodel.seqloc

Examples

data(NCBIeaa)
ncbiTaxonomy

**ncbiTaxonomy**

**NCBI Taxonomy Database**

**Description**

Retrieve taxonomic classifications from the taxonomy database at the NCBI.

**Usage**

```r
ncbiTaxonomy(taxon, species.list = FALSE, kingdom, trim, parallel = FALSE, type = "SOCK")
```

**Arguments**

- **taxon**
  - a character string giving the name of the clade of interest.

- **species.list**
  - logical: is tax list of species or a set of higher taxon names?

- **kingdom**
  - a character string giving the kingdom to which term belongs. This in information is sometimes necessary as the lineages in different kingdoms can have identical names (e.g., *Glaucidium* in Metazoa and Viridiplantae) and the value of kingdom is then used for disambiguation.

- **trim**
  - a character string determining if and how to drop uninformative higher ranks. The default, `trim = "auto"`, drops all ranks higher than the rank of term. Alternatively, any existing name of a higher rank (e.g., "family", "kingdom", ...) may be given.

- **parallel**
  - logical, indicating if `ncbiTaxonomy` should be executed serial or parallel

- **type**
  - character

**Value**

a data frame containing the taxonomic classification of all species belonging to term, for which molecular sequence information is found at the NCBI.

**Author(s)**

Christoph Heibl

**References**


**See Also**

- `dbUpdateTaxonomy` and `dbReadTaxonomy` for storing and retrieving taxonomic information in a *megaptera* project database.
prune.phylo.rank  

**Prune Phylogenies**

**Description**

Prune tips of a certain taxonomic rank from an object of class `phylo` and obtain a new `phylo` object whose tips are of a higher rank.

**Usage**

prune.phylo.rank(phy, tax, rank = "gen", add.nb = TRUE, quiet = FALSE)

**Arguments**

- `phy`: an object of class `phylo`.
- `tax`: a data frame containing taxonomic information for the tip labels in `phy`.
- `rank`: a character string giving the name of a column (= taxonomic rank) in `tax` to which `phy` will be pruned.
- `add.nb`: logical
- `quiet`: logical

**Value**

an object of class `phylo`.

**Author(s)**

Christoph Heibl

**See Also**

- `addTip` to add terminal nodes to a phylogeny.

---

**setLocus**  

**Set locus in megapteraProj**

**Description**

Sets the `locus` slot in `megapteraProj`.

**Usage**

setLocus(x, locus)
**stepA**

**Arguments**
- `x` an object of class `megapteraProj`.
- `locus` an object of class `locus`.

**Value**
- an object of class `megapteraProj`.

**Author(s)**
Christoph Heibl

**See Also**
`megapteraProj`, `locus`.

---

**Description**
Download a taxonomic classification of the focal taxa from the NCBI taxonomy database and store it in the PostgreSQL database.

**Usage**
```r
stepA(x)
```

**Arguments**
- `x` an object of class `megapteraProj`

**Value**
None. `stepB` is called for its side effects: (1) a taxonomic classification is stored in the PostgreSQL database; (2) a log file is written to the current working directory.

**Author(s)**
Christoph Heibl

**See Also**
`megapteraProj`
Step B: Search and Download Sequences

Description

For any given project (see `megapteraProj`), this step searches the Nucleotide database on GenBank, downloads all sequences and stores them in a PostgreSQL database table.

Usage

`stepB(x)`

Arguments

- `x` an object of class `megapteraProj`

Details

All accessions are stored under their species name as appearing in the `organism` field at GenBank, information about infrageneric ranks is stripped of the taxon names before they are stored in the database.

Value

None. `stepB` is called for its side effects: (1) strings of DNA sequences with attribute data are stored in a pgSQL database; (2) a log file is written to the current working directory.

Note

The function is under development and the results should always be checked.

Author(s)

Christoph Heibl

References


See Also

`megapteraProj`
Step C: Align Conspecific Sequences

Description
Align conspecific sequences. In this context, ‘conspecific’ refers to sequences whose names as given by the organism field spell absolutely identical.

Usage
stepC(x)

Arguments
x an object of class megapteraProj

Note
The function is under development and the results should always be checked.

Author(s)
Christoph Heibl

See Also
megapteraProj; stepB for the preceeding and stepD for the subsequent step.

Step D: Calculate Benchmark Sequences

Description
In order to evaluate sequence identity and quality, each sequence will be compared to a ‘benchmark sequence’, which is constructed in stepD.

Usage
stepD(x)

Arguments
x an object of class megapteraProj.

Note
The function is under development and the results should always be checked.
Author(s)

Christoph Heibl

See Also

megapteraProj; stepC for the preceding and stepE for the subsequent step.

---

**Step E: Calculate Each Accession’s Distance from Benchmark**

**Description**

In stepE the genetic distance of each sequence from the corresponding benchmark sequence will be calculated and written to the PostgreSQL database.

**Usage**

stepE(x)

**Arguments**

x an object of class megapteraProj.

**Note**

The function is under development and the results should always be checked.

Author(s)

Christoph Heibl

See Also

megapteraProj; stepD for the preceding and stepF for the subsequent step.
**Step F: Select Sequences and Assemble FASTA file**

**Description**
In step F, FASTA files will be assembled selecting all sequences that passed the quality evaluation.

**Usage**
stepF(x)

**Arguments**

- **x**
  an object of class `megapteraProj`.

**Note**
The function is under development and the results should always be checked.

**Author(s)**
Christoph Heibl

**See Also**
- `megapteraProj; stepE` for the preceding and `stepG` for the subsequent step.

---

**Step G: Profile Alignment**

**Description**
Performs a profile alignment (with MAFFT) of the sequences selected by `stepD` along a taxonomy built with `stepA`.

**Usage**
stepG(x, nob = FALSE)

**Arguments**

- **x**
  an object of class `megapteraProj`.
- **nob**
  logical

**Note**
The function is under development and the results should always be checked.
**Author(s)**

Christoph Heibl

**See Also**

`megapteraProj; stepF` for the preceding and `stepH` for the subsequent step.

---

**stepH**

*Step H: Detect and Separate Unalignable Blocks*

**Description**

Dependent on the substitution rate of the genomic region and the taxonomic depth of the study group, not always all accession can be aligned into a single alignment. *step H* will detect and separate such unalignable alignment blocks.

**Usage**

`stepH(x, clean = TRUE)`

**Arguments**

- `x` an object of class `megapteraProj`.
- `clean` logical: do cleaning/masking of alignment blocks?

**Value**

None. `stepA` is called for its side effects

**Note**

The function is under development and the results should always be checked.

**Author(s)**

Christoph Heibl

**See Also**

`megapteraProj; stepF` for the preceding step and `supermatrix` for the concatenation of loci.
Description

Delete infraspecific names and epithets from species names.

Usage

\[
\text{strip.spec}(x)
\]

\[
\text{strip.infraspec}(x, \text{force.underscore} = \text{FALSE})
\]

Arguments

- **x**: a character string
- **force.underscore**: logical, if TRUE, white space will be replaced by underscore

Details

\text{strip.spec} and \text{strip.infraspec} try to guess the separating character used; both " " and "_" are possible.

Value

a character string

Author(s)

Christoph Heibl

Examples

```r
(name <- "Vipera aspis aspis")
(name2 <- strip.infraspec(name))
(name3 <- strip.spec(name2))

names.set <- c("Vipera aspis aspis",
                "Vipera_aspis_atra")
sapply(names.set, strip.infraspec)
sapply(names.set, strip.infraspec, force.underscore = TRUE)
```
Description

Creates a S4 class defining a taxonomic parameters for a megapteraProj.

Usage

taxon(ingroup, outgroup, kingdom, hybrids = FALSE, reference.rank = "auto")

Arguments

- **ingroup**: a vector of mode "character", giving species names or names of higher taxa that define the focal group.
- **outgroup**: a vector of mode "character", giving species names or names of higher taxa that define the outgroup.
- **kingdom**: a vector of mode "character", currently one of "Fungi", "Metazoa", or "Viridiplantae". The International Code of Nomenclature for algae, fungi, and plants (ICN) and the International Code of Zoological Nomenclature (ICZN) do not exclude the possibility that genera and species of plants/fungi/algae and animals share the same name (e.g., Prunella vulgaris). Therefore it is necessary to include the kingdom into the search term when querying GenBank.
- **hybrids**: logical: if TRUE, hybrids (as recognized by the regular expression "^x_|_x_") will be excluded from the pipeline.
- **reference.rank**: a vector of mode "character", giving the name of the rank to be used to create subsets of the sequences to derive the reference sequence(s). The default ("auto") commits the selection of the reference rank to the pipeline and in most cases you should be fine using this option.

Value

an object of class taxon

Author(s)

Christoph Heibl

See Also

dbPars, locus, and megapteraPars for defining of database parameters, loci, and the pipeline’s parameters, respectively, and megapteraProj for the bundling of input data.
**Description**

Class for taxonomic input parameters for `megapteraProj`.

**Objects from the Class**

Objects can be created by calls of the form `taxon(ingroup, outgroup, kingdom, hybrids = FALSE, reference.rank)`.

**Slots**

- **ingroup** a vector of mode "character", giving species names or names of higher taxa that define the focal group.
- **outgroup** a vector of mode "character", giving species names or names of higher taxa that define the outgroup.
- **kingdom** a vector of mode "character", currently one of "Fungi", "Metazoa", or "Viridiplantae". The International Code of Nomenclature for algae, fungi, and plants (ICN) and the International Code of Zoological Nomenclature (ICZN) do not exclude the possibility that genera and species of plants/fungi/algae and animals share the same name (e.g., *Prunella vulgaris*). Therefore it is necessary to include the kingdom into the search term when querying GenBank.
- **species.list** logical indicating if slot ingroup is a list of species or a higher rank.
- **hybrids** logical: if TRUE, hybrids (as recognized by the regular expression "^x_|_x_") will be excluded from the pipeline.
- **reference.rank** a vector of mode "character", giving the name of the rank to be used to create subsets of the sequences to derive the reference sequence(s). The default ("auto") commits the selection of the reference rank to the pipeline and in most cases you should be fine using this option.

**Methods**

- **show** signature(object = "taxon"): prints taxonomic parameter setting

**Author(s)**

Christoph Heibl

**See Also**

`dbPars`, `locus`, and `megapteraPars` for defining of database parameters, loci, and the pipeline's parameters, respectively, and `megapteraProj` for the bundling of input data.
Index

*Topic classes
dbPars-class, 6
taxon-class, 27
*Topic datasets
cetacea, 4
NCBIeaa, 16
*Topic package
megaptera-package, 2
addTip, 3, 9, 18
ape, 3
cetacea, 4
check.Clades (check Functions), 5
check.Coverage (check Functions), 5
check.Functions, 5
check.Markers (check Functions), 5
check.NameChanges (check Functions), 5
check.Species (check Functions), 5
dbConnect, 7–9
dbPars, 6, 7, 10, 11, 14, 15, 26, 27
dbPars-class, 6
dbReadDNA, 8
dbReadDNA (io.dna.db), 9
dbReadTaxonomy, 7, 9, 17
dbUpdateTaxonomy, 8, 17
dbWriteDNA, 8
dbWriteDNA (io.dna.db), 9
DNAbin, 9
fleshoutGuidetree, 8
gblocks, 12, 13
io.dna.db, 9
locus, 6, 7, 10, 14, 15, 19, 26, 27
locus-class, 11
megaptera-package, 2
megapteraPars, 6, 7, 10, 11, 14, 15, 26, 27
megapteraPars-class, 13
megapteraProj, 6, 7, 10–14, 18–24, 26, 27
megapteraProj-class, 15
NCBIeaa, 16
ncbiTaxonomy, 8, 17
phylo, 3, 5, 9, 18
prune.phylo.rank, 3, 18
regex, 9
setLocus, 18
stepA, 19
stepB, 20, 21
stepC, 21, 22
stepD, 21, 21, 22, 23
stepE, 22, 22, 23
stepF, 22, 23, 24
stepG, 23, 23
stepH, 24, 24
strip.infraspec, 25
strip.spec (strip.infraspec), 25
supermatrix, 24
taxon, 6, 10, 11, 14, 15, 26
taxon-class, 27