Package 'phylogenize'

June 13, 2019

Title Associate Microbial Prevalence and Specificity with Gene Presence

Version 0.0.0.9200

Description phylogenize contains functions to estimate microbial prevalence and specificity scores from data, to associate these quantitative phenotypes with gene presence/absence (using the implementation of phylogenetic regression from the package phylolm), and to visualize the results. Both shotgun metagenomics and 16S amplicon data can be used with this pipeline.

License MIT

Encoding UTF-8

LazyData true

Depends phylolm,
settings,
Matrix,
tidyverse,
ggtree,
biomformat,
methods,
stats,
graphics,
grDevices,
functional,
future,
furrr

Imports data.table,
parallel,
qvalue,
ape,
phytools,
knitr,
kableExtra,
scales,
xml2,
ggplot2,
MASS,
seqinr,

pbapply,
gtools,
svglite,
pryr,
testthat,
ezknitr

Suggests

RoxygenNote 6.1.1.9000

R topics documented:

`add.below.LOD` *Add in taxa that were not observed, assuming this means they were zero-prevalence.*

Description

Add in taxa that were not observed, assuming this means they were zero-prevalence.

Usage

```
add.below.LOD(pz.db, abd.meta, ...)
```

Arguments

`pz.db`	A database (typically obtained with `import.pz.db`).
`abd.meta`	A list consisting of a taxon abundance matrix and the metadata.

Value

An updated version of `abd.meta`.

add.sig.descs *Add gene descriptions to significant results; return in a tibble.*

Description

Add gene descriptions to significant results; return in a tibble.

Usage

```
add.sig.descs(phy.with.sigs, pos.sig, gene.to.fxn)
```

Arguments

phy.with.sigs	Character vector giving the phyla with significant hits.
pos.sig	List of character vectors, one per phylum, of significant hits.
gene.to.fxn	Data frame used to annotate genes to functions.

Value

A single data frame of all significant results plus descriptions.

adjust.db *Clean up imported database.*

Description

adjust.db removes any phyla with fewer than opts('treemin') representatives, removes tips from trees that were not observed in the data, and resolves polytomies. It also adds a couple of variables into the database that give lists of taxa per tree and the total number of remaining phyla.

Usage

```
adjust.db(pz.db, abd.meta, ...)
```

Arguments

pz.db	A database (typically obtained with import.pz.db).
abd.meta	A list consisting of a taxon abundance matrix and the metadata.

Value

An updated database.

`alt.multi.enrich` *An alternative way to get the enrichment table, using a tbl of results instead of separate inputs for significant results, effect signs, etc.*

Description

An alternative way to get the enrichment table, using a tbl of results instead of separate inputs for significant results, effect signs, etc.

Usage

```
alt.multi.enrich(results, mappings, dirxn = 1, qcuts = c(strong = 0.05,
  med = 0.1, weak = 0.25), future = FALSE)
```

Arguments

`results`	A tidyverse tbl of results with at least the following columns: "phylum", "gene", "effect.size", and "q.value" (if only "p.value" is present, first apply function `result.qvs`).
`mappings`	List of data.frames giving gene-to-gene-set mappings.
`dirxn`	Count only genes with this effect sign as significant.

Value

A tbl giving Fisher's test p-values, q-values, effect sizes, and overlaps.

`calc.alpha.power` *Calculate the FPR and (1 - FNR) for results of a set of tests.*

Description

Calculate the FPR and (1 - FNR) for results of a set of tests.

Usage

```
calc.alpha.power(pvs, null, alt, alpha = 0.05, filter = NULL)
```

Arguments

`pvs`	A named vector of p-values, one per test.
`null`	A vector of strings giving the tests in `pvs` for which the null was true.
`alt`	A vector of strings giving the tests in `pvs` for which the alternative hypothesis was true.
`filter`	Optional vector of strings giving the tests to which the analysis should be restricted.

Value

A numeric vector:

r	Proportion of p-values where the null was rejected.
p	Power (1 - FNR)
a	Alpha (FPR)

`calc.ess` *Master function to calculate environmental specificity scores.*

Description

Some particularly relevant global options are:

env_column String. Name of column in metadata file containing the environment annotations.

dset_column String. Name of column in metadata file containing the dataset annotations.

which_envir String. Environment in which to calculate prevalence or specificity. Must match annotations in metadata.

prior_type String. What type of prior to use ("uninformative" or "file").

Usage

```
calc.ess(abd.meta, pdata = NULL, b.optim = NULL, ...)
```

Arguments

abd.meta	A list giving an abundance matrix and metadata.
pdata	Named numeric vector giving priors per environment.
b.optim	If not NULL, use this value for the regularization parameter b, otherwise optimize it.

Value

An additively-smoothed estimate of taxon prevalences.

check.process.metadata

Check and process metadata

Description

`check.process.metadata` is used to make sure that the metadata satisfies the requirements specified by the global options and to make sure that the metadata are of the correct type.

Usage

```
check.process.metadata(metadata, ...)
```

Arguments

`metadata`	A data frame of metadata with environment, dataset, and sample columns corresponding to those in the global options (see \?pz.options).

Details

Some particularly relevant global options are:

env_column Name of metadata column containing environment annotations.

dset_column Name of metadata column containing dataset annotations.

single_dset Boolean. If true, will assume that all samples come from a single dataset called `dset1` no matter what, if anything, is in `dset_column`.

Value

A data frame of metadata, with environment and dataset columns converted to factors.

clean.pheno

Remove taxa from a phenotype that aren't in our trees and gene matrices (usually only necessary for testing).

Description

Remove taxa from a phenotype that aren't in our trees and gene matrices (usually only necessary for testing).

Usage

```
clean.pheno(phenotype, pz.db)
```

Arguments

phenotype	A quantitative phenotype (named numeric vector).
pz.db	A database.

Value

A phenotype with only the measurements represented in the database.

---

do.clust.plot *Make a hybrid tree-heatmap plot showing the taxon distribution of significant hits.*

Description

This function wraps single.cluster.plot, running it in a separate process. This is because there can be problems with memory leaks. For relevant global options, see the doumentation for that function.

Usage

```
## S3 method for class 'clust.plot'
do(gene.presence, sig.genes, tree, plotted.tree,
  phylum, verbose = FALSE, ...)
```

Arguments

gene.presence	Gene presence/absence matrix.
sig.genes	Character vector of the significant genes.
tree	A tree object.
plotted.tree	A ggtree plot of tree.
phylum	Name of the phylum represented by tree
verbose	Whether to report debugging information (boolean).

---

do.fisher *Wrapper around* fisher.test *that starts with string vectors instead of a contingency table.*

Description

Wrapper around fisher.test that starts with string vectors instead of a contingency table.

Usage

```
## S3 method for class 'fisher'
do(list1, list2, background, alt = "two.sided")
```

Arguments

list1	A vector of strings (e.g., names of significant gene hits).
list2	A vector of strings (e.g., names of genes in a pathway).
background	A vector of strings that list1 and list2 were drawn from (e.g., names of all genes tested in an assay). Any strings in list1 or list2 not in this vector will be discarded.
alt	The alternative hypothesis (see ?fisher.test).

Value

The results of fisher.test on a contingency table generated from list1, list2, and background, augmented with the additional field overlap which gives the intersection of list1 and list2.

---

gene.annot *Annotate genes using a gene-to-function table.*

Description

Annotate genes using a gene-to-function table.

Usage

```
gene.annot(x, gene.to.fxn)
```

Arguments

x	A gene (string) or vector of genes (strings).
gene.to.fxn	A data frame with at least "gene" and "function" as columns.

Value

A character vector of gene functions, with names equal to x.

get.burst.results *Read in results from BURST.*

Description

get.burst.results reads and parses the output of BURST to get the best-hit MIDAS species identifier for any 16S hit. Note that the reference 16S FASTA database file must describe entries in the format: ">gene species_or_genus_ID MIDAS_ID". Only MIDAS_ID is used so the contents of "gene" and "species_or_genus_ID" can be arbitrary.

Usage

```
get.burst.results(...)
```

Details

Some particularly relevant global options are:

in_dir String. Path to input directory (i.e., where to look for input files).

burst_outfile String. File name where BURST writes output which is then read back into *phylogenize*.

Value

List containing a vector of hits, a vector of MIDAS ID targets, and a data frame of the assignments as they came out of BURST.

get.pheno.plotting.scales *Get phenotype plotting scales.*

Description

Some particularly relevant global options are:

which_phenotype String. Which phenotype to calculate ("prevalence" or "specificity").

prev_color_low String. When graphing prevalence on a tree, this color is the lowest value.

prev_color_high String. When graphing prevalence on a tree, this color is the highest value.

spec_color_high String. When graphing specificity on a tree, this color is the lowest value (most anti-specific).

spec_color_med String. When graphing specificity on a tree, this color denotes the prior (no association).

spec_color_high String. When graphing specificity on a tree, this color is the highest value (most specific).

Usage

```
get.pheno.plotting.scales(phenotype, trees, phenoP = NULL, ...)
```

Arguments

phenotype	A named vector giving the phenotype for each taxon ID.
trees	A list of tree objects.
phenoP	An optional value giving the prior probability for the environment of interest.

Value

A list of overall limits (limits), phylum-specific limits (phy.limits), a color scale (colors), and the zero point (zero).

---

get.top.N *Return the significant hits with the N smallest p-values.*

Description

Return the significant hits with the N smallest p-values.

Usage

```
get.top.N(p, sigs, signs, results, level = "strong", exclude = NULL,
  N = 25, total.n.cutoff = 0, genomes.per.protein = NULL)
```

Arguments

p	A phylum
sigs	The output of make.sigs.
signs	The output of make.signs.
results	List of result matrices, one per phylum.
level	Significance level (must be in sigs[[1]]).
exclude	Optional: exclude these genes from any list.
N	Integer; how many hits to return.
total.n.cutoff	Optional: if genomes.per.protein provided, only return hits found in at least this many genomes.
genomes.per.protein	Optional: list (one per phylum) of named numeric vectors giving the number of genomes that each protein was found in.

Value

A named vector of N significant hits in descending order of significance.

gg.cont.tree *Plot continuous trait on a tree.*

Description

gg.cont.tree paints a continuous trait along a tree.

Usage

```
gg.cont.tree(phy, ctrait, cAnc = NULL, model = "ARD",
  cLimits = logit(c(0.025, 0.1)), n = NULL, reduced.phy = NULL,
  colors = c(low.col = "slateblue", mid.col = "black", high.col =
  "orange2"), plot = T, restrict = NULL, cName = "prevalence",
  reverse = F, ladderize = T, ...)
```

Arguments

phy	A phylo object.
ctrait	A named numeric vector assigning trait values to tree tips.
cAnc	Calculated ancestry for continuous trait; if this is NULL, it is calculated.
model	Model for calculating ancestry (see phytools::fastAnc).
cLimits	Scale bar limits for plotting continuous trait.
n	Character vector giving which nodes to display; if NULL, defaults to intersection of phy$tip.label and names(ctrait).
reduced.phy	Dichotomous tree with only the nodes in n represented; if NULL, this is calculated.
colors	Named character vector with at least "low.col" and "high.col" and optionally "mid.col" defined, giving colors to use for plotting.
plot	Whether to plot the tree object or just return it.
restrict	Character vector giving which continuous trait values to plot; if NULL, all are used.
cName	String giving the title of the plot.
reverse	Mirror the resulting plotted tree.
ladderize	Ladderize the plotted tree.
...	Additional arguments passed to ggplot2.

Value

A ggtree plot of a continuous trait plotted along a tree.

hack.tree.labels *XML hack to make interactive tree diagrams.*

Description

This hack is very ugly but works most of the time. However, it is a good idea to wrap it in a tryCatch so that you can fall back to a less flashy implementation, because it relies on editing a poorly-annotated SVG file as if it were an XML document.

Usage

```
hack.tree.labels(tree.obj, file, stroke.scale = 0.7, pheno = NULL,
  pheno.name = NULL, native.tooltip = FALSE, units = "", ...)
```

Arguments

tree.obj	A ggtree representation of a tree.
file	A filename where the final SVG output will be written.
stroke.scale	Multiplier of stroke width in dendrogram.
pheno	A vector with names corresponding to the tips of the tree and values corresponding to the phenotype value at that tip.
pheno.name	The name of the phenotype being calculated (e.g. "prevalence").
native.tooltip	Instead of using mouseover, use SVG tooltips (less powerful).
units	Postfix for the values in pheno (e.g. " percentages).

harmonize.abd.meta *Check metadata and abundance matrix against one another*

Description

harmonize.abd.meta compares the abundance matrix to the metadata matrix to make sure that enough samples are in common between the two to perform an *phylogenize* analysis, after dropping any singleton datasets or environments (effects for these cannot be estimated).

Usage

```
harmonize.abd.meta(abd.meta, ...)
```

Arguments

abd.meta	A list with components mtx and metadata, corresponding to a sparse binary presence/absence matrix (see Matrix package) and a metadata data frame.

Details

Some particularly relevant global options are:

env_column Name of metadata column containing environment annotations.

dset_column Name of metadata column containing dataset annotations.

Value

A list of the same form as `abd.meta`.

`import.pz.db` *Import the data necessary for *phylogenize* analysis.*

Description

`import.pz.db` decides based on global options which data files to import.

Usage

```
import.pz.db(...)
```

Details

Some particularly relevant global options are:

type String. Type of data to use, either "midas" (shotgun) or "16S" (amplicon).

db_version String. Which version of the MIDAS database to use ("midas_v1.2" or "midas_v1.0").

data_dir String. Path to directory containing the data files required to perform a *phylogenize* analysis.

Value

A list of the data objects required to perform a *phylogenize* analysis, with components `gene.presence`, `trees`, `phyla`, `taxonomy`, `g.mappings`, and `gene.to.fxn`.

install.data.figshare *Download data from figshare (or provide it locally) and un-gzip it into the package directory so that it can be imported.*

Description

Download data from figshare (or provide it locally) and un-gzip it into the package directory so that it can be imported.

Usage

```
install.data.figshare(data_path = NULL,
 figshare_url = "https://ndownloader.figshare.com/files/15013790?private_link=122ea0030cf11c65e32b")
```

Arguments

data_path	Optional: provide a path to a local file containing a compressed .tar archive of data. Must extract to the subdirectory `extdata/`.
figshare_url	Optional: override the URL from which to obtain the data.

kable.recolor *Function to obtain HTML colors for a particular value.*

Description

Function to obtain HTML colors for a particular value.

Usage

```
kable.recolor(x, direction = 1, option = "D", na_color = "#BBBBBB",
  scale_from = NULL, colors = c("#000000", "#FFFFFF"),
  limits = c(-Inf, Inf))
```

Arguments

x	A value or vector of values to colorize.
direction	If 1, use the color scale as given; if -1, reverse it.
na_color	Set NA elements to this color.
scale_from	Instead of the minimum and maximum of `x`, scale values from this minimum and maximum (see `?rescale`).
colors	A vector of two strings giving the low and high color, respectively.
limits	A vector of two numbers giving the minimum and maximum value outside which values will be represented by the bottom or top of the color scale, respectively.

Value

An HTML color.

`keep.tips` *Keep some tips on a tree.*

Description

`keep.tips` keeps only the set of specified tips in a tree.

Usage

```
keep.tips(tree, keep)
```

Arguments

`tree`	A `phylo` object.
`keep`	A character vector of tip labels. Any tip not in this vector will be dropped.

`lm.fx.pv` *Wrapper around* lm *that returns just the effect size and p-value.*

Description

Wrapper around *lm* that returns just the effect size and p-value.

Usage

```
lm.fx.pv(m, p, tr, coefname = "mTRUE", restrict = NULL,
  meas_err = FALSE)
```

Arguments

`m`	Named numeric vector of gene presence/absences per taxon.
`p`	Named numeric vector of phenotype values per taxon.
`tr`	Phylogeny relating taxa (class `"phylo"`).

Value

Length-2 numeric vector with names `"Estimate"` and `"p.value"`. If there is an error in `phylolm`, the values of this vector will be `c(NA,NA)`.

logistic *Calculate inverse-logit of a value or vector of values.*

Description

Calculate inverse-logit of a value or vector of values.

Usage

```
logistic(x)
```

Arguments

x	Numeric value, or numeric vector of numeric values.

logit *Calculate logit of a value or vector of values.*

Description

Calculate logit of a value or vector of values.

Usage

```
logit(x)
```

Arguments

x	Numeric value, or numeric vector of numeric values.

make.pos.sig *Get vectors of significant genes with positive effect sizes.*

Description

Get vectors of significant genes with positive effect sizes.

Usage

```
make.pos.sig(sigs, signs, cut = "strong")
```

Arguments

sigs	Output of make.sigs
signs	Output of make.signs
cut	String giving named significance level to use.

Value

List (per phylum) of string vectors of positive significant hits.

---

make.results.matrix *Convert results into a long (vs. wide) format.*

Description

Convert results into a long (vs. wide) format.

Usage

```
make.results.matrix(results)
```

Arguments

results	Output of result.wrapper.plm.

Value

A single data frame with entries from results.

---

make.signs *Get effect sizes of genes from result tables.*

Description

Get effect sizes of genes from result tables.

Usage

```
make.signs(results)
```

Arguments

results	List of result matrices with two rows (effect size and p-value) and one column per gene tested.

Value

List (per phylum) of numeric vectors of signs of hits.

make.sigs *Get vectors of significant genes from result tables.*

Description

Get vectors of significant genes from result tables.

Usage

```
make.sigs(results, cuts = c(strong = 0.05, med = 0.1, weak = 0.25),
  method = qvals, exclude = NULL, min.fx = 0)
```

Arguments

results	List of result matrices with two rows (effect size and p-value) and one column per gene tested.
cuts	Named numeric vector giving different significance cutoffs.
method	Function that will be used to adjust raw p-values in results.
exclude	String vector of genes to exclude (optional).
min.fx	Minimum effect size for calling something significant.

Value

List (per phylum) of string vectors of significant hits.

matrix.plm *Perform phylogenetic (or linear) modeling for a single phylum.*

Description

Perform phylogenetic (or linear) modeling for a single phylum.

Usage

```
matrix.plm(tree, mtx, pheno, method = phylolm.fx.pv,
  restrict.taxa = NULL, restrict.ff = NULL, ...)
```

Arguments

tree	A tree relating taxa within a phylum.
mtx	Gene presence/absence matrix.
pheno	Named numeric vector giving phenotype values per taxon.
method	A function that returns a length-2 numeric vector of effect-size and p-value (see, e.g., phylolm.fx.pv or lm.fx.pv).
restrict.taxa	Optionally, a character vector giving a subset of taxa to test.
restrict.ff	Optionally, a character vector giving a subset of genes to test.

Value

Matrix of p-values (row 1) and effect-sizes (row 2) per gene (columns).

`multi.enrich` *Given lists of significant genes (at different thresholds), effect sizes, and gene set mappings, assemble a tbl of results.*

Description

Given lists of significant genes (at different thresholds), effect sizes, and gene set mappings, assemble a tbl of results.

Usage

```
multi.enrich(sigs, signs, mappings, dirxn = 1)
```

Arguments

`sigs`	List giving, per phylum (outer) and per significance cutoff (inner), significant hits to test for enrichment.
`signs`	List giving, per phylum, signs of all gene effect sizes.
`mappings`	List of data.frames giving gene-to-gene-set mappings.
`dirxn`	Count only genes with this effect sign as significant.

Value

A tbl giving Fisher's test p-values, q-values, effect sizes, and overlaps.

`non.interactive.plot` *A fall-back plotting option for when* `hack.tree.labels` *fails, designed to produce the same kind of output.*

Description

A fall-back plotting option for when `hack.tree.labels` fails, designed to produce the same kind of output.

Usage

```
non.interactive.plot(tree.obj, file)
```

Arguments

`tree.obj`	A ggtree object.
`file`	File to which an SVG representation of this tree object will be written.

nonequiv.pos.sig *Get vectors of significant genes with positive effect sizes.*

Description

Get vectors of significant genes with positive effect sizes.

Usage

```
nonequiv.pos.sig(results, method = qvals, qcut_sig = 0.05,
  qcut_eq = 0.05, min_fx = 0.25, exclude = NULL)
```

Arguments

results	List of result matrices (4 x N).
method	Method for performing multiple test adjustment.
qcut_sig	Desired q-value cutoff for significance test.
qcut_eq	Desired q-value cutoff for equivalence test (N.B.: significantly equivalent hits will be *excluded*).
min_fx	Minimum effect size for equivalence test.
exclude	Optional list of character vectors of genes to exclude.

Value

List of character vectors of hits that were significantly different from zero, had positive effect sizes, and not significantly equivalent to a minimum effect size.

nonparallel.results.generator *Fit phylogenetic (or linear) models (single core version, single phylum).*

Description

Fit phylogenetic (or linear) models (single core version, single phylum).

Usage

```
nonparallel.results.generator(gene.matrix, tree, taxa, pheno,
  phylum.name = "TestPhylum", method = phylolm.fx.pv,
  restrict.ff = NULL, remove.low.variance = TRUE,
  use.for.loop = TRUE, ...)
```

Arguments

gene.matrix	Gene presence/absence matrix.
tree	Phylogeny relating taxa.
pheno	Named numeric vector giving phenotype values per taxon.
phylum.name	Name of phylum being considered.
method	A function that returns a length-2 numeric vector of effect-size and p-value (see, e.g., phylolm.fx.pv or lm.fx.pv).
restrict.ff	Optionally, a character vector giving a subset of genes to test.
remove.low.variance	Boolean giving whether to drop genes that are always present or always absent in a particular phylum.
use.for.loop	Boolean giving whether to use a for loop instead of a pbapply.

Value

Named list of p-value and effect-size matrices, one per phylum.

---

output.enr.table *Make a pretty enrichment table.*

Description

Make a pretty enrichment table.

Usage

```
output.enr.table(enr.table)
```

Arguments

enr.table	Input enrichment table.

Value

Mutated enrichment table with better-labeled columns and significance coloring.

`pheno_nonzero_var` *Return a boolean telling whether a phenotype has nonzero variance in different phyla.*

Description

Return a boolean telling whether a phenotype has nonzero variance in different phyla.

Usage

```
pheno_nonzero_var(phenotype, taxa)
```

Arguments

`phenotype`	A quantitative phenotype.
`taxa`	From ‘pz.db$taxa‘.

Value

A boolean vector with length equal to ‘length(taxa)‘.

`phylolm.fx.pv` *Wrapper around* phylolm *that returns just the effect size and p-value.*

Description

Wrapper around *phylolm* that returns just the effect size and p-value.

Usage

```
phylolm.fx.pv(m, p, tr, coefname = "mTRUE", restrict = NULL,
  meas_err = FALSE)
```

Arguments

`m`	Named numeric vector of gene presence/absences per taxon.
`p`	Named numeric vector of phenotype values per taxon.
`tr`	Phylogeny relating taxa (class `"phylo"`).
`coefname`	Which coefficient from the phylolm to return?
`restrict`	If not NULL, a character vector of taxa to consider.

Value

Length-2 numeric vector with names `"Estimate"` and `"p.value"`. If there is an error in `phylolm`, the values of this vector will be `c(NA,NA)`.

`plot.labeled.phenotype.trees`
Edit a list of plotted trees to add fancy highlight labels.

Description

This function adds fancy SVG highlight labels to ggtree objects and then plots them. If there's an error, it will fall back to a regular plot.

Usage

```
## S3 method for class 'labeled.phenotype.trees'
plot(plotted.pheno.trees, phenotype,
  label = "prevalence", stroke.scale = 0.3, units = "%")
```

Arguments

`plotted.pheno.trees`	A named list of ggtree plots (per phylum).
`phenotype`	Phenotype to plot/label.
`label`	Label to give to the phenotype.
`stroke.scale`	How thick to make the highlight.
`units`	A string appended to each label, used to give units of phenotype.

Details

Some particularly relevant global options are:

which_phenotype String. Which phenotype to calculate ("prevalence" or "specificity").

`plot.pheno.distributions`
Plot distributions of a phenotype across phyla.

Description

Some particularly relevant global options are:

which_phenotype String. Which phenotype to calculate ("prevalence" or "specificity").

Usage

```
## S3 method for class 'pheno.distributions'
plot(phenotype, pz.db, ...)
```

Arguments

phenotype	A named vector with the phenotype values for each taxon.
pz.db	A database containing a taxonomy and trees.

Value

A ggplot object with the phenotype distribution plotted per phylum.

---

plot.phenotype.trees *Plot a phenotype along a list of trees.*

Description

Some particularly relevant global options are:

which_phenotype String. Which phenotype to calculate ("prevalence" or "specificity").

Usage

```
## S3 method for class 'phenotype.trees'
plot(phenotype, trees, scale, ...)
```

Arguments

phenotype	A named vector with the phenotype values for each taxon.
trees	A list of trees.
scale	A list returned from get.pheno.plotting.scales.

Value

A list of ggtree objects in which the phenotype has been plotted across each tree in trees.

---

prepare.burst.input *Prepare input file for BURST analysis.*

Description

prepare.burst.input outputs a FASTA file of the sequences in the input 16S data for analysis using BURST.

Usage

```
prepare.burst.input(mtx, ...)
```

Arguments

mtx	A presence/absence or abundance matrix, with row names equal to amplicon sequence variant DNA sequences.

Details

Some particularly relevant global options are:

in_dir String. Path to input directory (i.e., where to look for input files).

burst_infile String. File name of the sequences written to disk and then read into BURST.

---

prev.addw *Master function to calculate taxon prevalences with additive smoothing.*

Description

Some particularly relevant global options are:

env_column String. Name of column in metadata file containing the environment annotations.

dset_column String. Name of column in metadata file containing the dataset annotations.

which_envir String. Environment in which to calculate prevalence or specificity. Must match annotations in metadata.

Usage

```
prev.addw(abd.meta, ...)
```

Arguments

abd.meta	A list giving an abundance matrix and metadata.

Value

An additively-smoothed estimate of taxon prevalences.

process.16s *Map denoised 16S sequence variants to MIDAS IDs using BURST.*

Description

process.16s is a wrapper for other functions that: output sequence variants to a file; map them using BURST against a reference database of 16S sequences; then return a list of abundance and metadata values where the rows of the abundance matrix are now MIDAS IDs.

Usage

```
process.16s(abd.meta, ...)
```

Arguments

mtx	A presence/absence or abundance matrix, with row names equal to amplicon sequence variant DNA sequences.

Details

Some particularly relevant global options are:

in_dir String. Path to input directory (i.e., where to look for input files).

burst_infile String. File name of the sequences written to disk and then read into BURST.

Value

none

pv1 *Fix p-values that are above 1.*

Description

Sometimes, p-values from the Fisher test can apparently be slightly larger than one for some reason; this works around that problem.

Usage

```
pv1(x)
```

Arguments

x	A vector of p-values.

Value

The same vector with all p-values above 1 changed to exactly 1.

pz.error *Throw an error and optionally log it in errmsg.txt.*

Description

Some particularly relevant global options are:

error_to_file Boolean. Should pz.error, pz.warning, and pz.message output to an error message file?

Usage

```
pz.error(errtext, ...)
```

Arguments

errtext	String: error message text.

pz.message *Report a message and optionally log it in errmsg.txt.*

Description

Some particularly relevant global options are:

error_to_file Boolean. Should pz.error, pz.warning, and pz.message output to an error message file?

Usage

```
pz.message(msgtext, ...)
```

Arguments

errtext	String: message text.

`pz.options` *Set and get options for phylogenize.*

Description

Function to set and get global options for the *phylogenize* package.

Usage

```
pz.options(...)
```

Arguments

`...`	Names of options (to retrieve) or `[key]=[value]` pairs (to set).

Details

These options are global because they affect how most of the functions in *phylogenize* work. Descriptions of these options follow.

File input/output and paths

out_dir String. Path to output directory. Default: "output"

in_dir String. Path to input directory (i.e., where to look for input files). Default: "."

data_dir String. Path to directory containing the data files required to perform a *phylogenize* analysis. Default: "./data", but on package load, this default is set to the result of `system.file("extdata",package="phylo`

working_dir String. Path to directory where relative paths should originate from. Default: `"."`

abundance_file String. Name of abundance tabular file. Default: "test-abundance.tab"

metadata_file String. Name of metadata tabular file. Default: "test-metadata.tab"

biom_file String. Name of BIOM abundance-and-metadata file. Default: "test.biom"

separate_metadata Boolean. For BIOM data, is there a separate tabular abundance table? Default: FALSE

input_format String. Whether to look for tabular or BIOM-formatted data ("tabular" or "biom"). Default: "tabular"

phenotype_file String. Name of input file for optional pre-calculated phenotype. Default: ""

prior_file String. File name of optional pre-computed prior. Default: ""

error_to_file Boolean. Should pz.error, pz.warning, and pz.message output to an error message file? Default: FALSE

biom_dir String. Path to BIOM executables. Only used during testing. Default: "/usr/local/bin/"

burst_dir String. Path where the binary of BURST is found. Default: "/usr/local/bin/"

burst_bin String. File name of the binary of BURST. Default: "burst12"

burst_16sfile String. Path to the 16S FASTA database that maps back to MIDAS species. Default: "16s_renamed.frn"

burst_infile String. File name of the sequences written to disk and then read into BURST. Default: "input_seqs.txt"

burst_outfile String. File name where BURST writes output which is then read back into *phylogenize*. Default: "output_assignments.txt"

Computing phenotypes

ncl Integer. Number of cores to use for parallel computation. Default: 1

type String. Type of data to use, either "midas" (shotgun) or "16S" (amplicon). Default: "midas"

env_column String. Name of column in metadata file containing the environment annotations. Default: "env"

dset_column String. Name of column in metadata file containing the dataset annotations. Default: "dataset"

sample_column String. Name of column in metadata file containing the sample IDs. Default: "sample_id"

single_dset Boolean. If true, will assume that all samples come from a single dataset called `dset1` no matter what, if anything, is in `dset_column`. Default: FALSE

db_version String. Which version of the MIDAS database to use ("midas_v1.2" or "midas_v1.0"). Default: "midas_v1.2"

which_phenotype String. Which phenotype to calculate ("prevalence" or "specificity"). Default: "prevalence"

which_envir String. Environment in which to calculate prevalence or specificity. Must match annotations in metadata. Default: "Stool"

prior_type String. What type of prior to use ("uninformative" or "file"). Default: "uninformative"

minimum Integer. A particular gene must be observed, and also absent, at least this many times to be reported as a significant positive association with the phenotype. Default: 3

assume_below_LOD Boolean. If TRUE, MIDAS species that are not present are assumed to have a prevalence of zero; if FALSE, they are dropped from the analysis. Default: TRUE

linearize Boolean. If TRUE, use a regular linear model instead of a phylogenetic linear model. Mostly useful for testing report generation, since the linear model is much faster but returns many more false positives. Default: FALSE

burst_cutoff Float. Value between 0.95 and 1.00 giving the percent ID cutoff to use when assigning denoised sequence variants to MIDAS species using BURST. Default: 0.985

meas_err Boolean. Separately estimate measurement error from phenotype variation in the phylogenetic linear model. Default: TRUE

min_fx Positive double. Effects that are significantly equivalent to this effect size will be excluded from significant positive hits. If zero, the equivalence test will be skipped. Default: 0

Graphing

treemin Integer. A phylum must have at least this many representatives in order to be graphed in the report. Default: 5

pctmin Integer. A phylum must have at least this percent of observed representatives in order to be graphed in the report. Default: 0.01

skip_graphs Boolean. If TRUE, skip making graphs in the report, which can be time- and memory-consuming. Default: FALSE

prev_color_low String. When graphing prevalence on a tree, this color is the lowest value. Default: "black"

prev_color_high String. When graphing prevalence on a tree, this color is the highest value. Default: "orange2"

spec_color_high String. When graphing specificity on a tree, this color is the lowest value (most anti-specific). Default: "slateblue"

spec_color_med String. When graphing specificity on a tree, this color denotes the prior (no association). Default: "gray50"

spec_color_high String. When graphing specificity on a tree, this color is the highest value (most specific). Default: "tomato"

gene_color_absent String. When graphing gene presence/absence, this color indicates absence. Default: "black"

gene_color_present String. When graphing gene presence/absence, this color indicates presence. Default: "black"

Memory management

pryr Boolean. If TRUE, report memory usage when generating the report. Default: FALSE

separate_process Boolean. When displaying clustered top gene associations alongside a tree colored by phenotype, this flag indicates whether to use a separate subprocess. This allows memory used by clustering to be released back to the operating system immediately. Default: TRUE

---

pz.warning *Report a warning and optionally log it in errmsg.txt.*

Description

Some particularly relevant global options are:

error_to_file Boolean. Should pz.error, pz.warning, and pz.message output to an error message file?

Usage

```
pz.warning(msgtext, ...)
```

Arguments

errtext	String: warning text.

read.abd.metadata *Read in abundance and metadata file(s).*

Description

Read in abundance and metadata, either as one BIOM-format file or as two tab-delimited files.

Usage

```
read.abd.metadata(...)
```

Details

This function uses package-wide options (see `?pz.options`), which can be overridden using the `...` argument. Some particularly relevant options are:

env_column String. Name of column in metadata file containing the environment annotations.

dset_column String. Name of column in metadata file containing the dataset annotations.

input_format String. Whether to look for tabular or BIOM-formatted data ("tabular" or "biom").

type String. Type of data to use, either "midas" (shotgun) or "16S" (amplicon).

Value

A list with components `mtx` and `metadata`, corresponding to a sparse binary presence/absence matrix (see `Matrix` package) and a metadata data frame.

remove.allzero.abundances *Remove rows and columns of a matrix that are all zero.*

Description

`remove.allzero.abundances` removes all rows and columns of a matrix where every observation is zero, starting with columns and then proceeding to rows.

Usage

```
remove.allzero.abundances(abd.mtx, ...)
```

Arguments

`abd.mtx`	A matrix of abundance values (double or logical).

Value

A matrix of abundance values (double), with all-zero columns and rows removed.

render.report *Run *phylogenize* start to finish.*

Description

Run *phylogenize* start to finish.

Usage

```
render.report(output_file = "report_output.html",
  report_input = "phylogenize-report.Rmd", do_cache = TRUE, ...)
```

Arguments

output_file	Path giving what to name the resulting HTML file.
report_input	Optionally override which notebook to knit (useful for testing).
do_cache	Turn on or off Rmarkdown's caching.
...	Parameters to override defaults.

result.wrapper.plm *Fit phylogenetic (or linear) models.*

Description

Fit phylogenetic (or linear) models.

Usage

```
## S3 method for class 'wrapper.plm'
result(phyla, pheno, tree, proteins, clusters,
  method = phylolm.fx.pv, restrict.figfams = NULL,
  drop.zero.var = FALSE, only.return.names = FALSE, ...)
```

Arguments

phyla	Character vector giving the names of the phyla.
pheno	Named numeric vector giving phenotype values per taxon.
tree	Either a single tree covering all taxa, or a list of per-phylum trees.
proteins	Named list of gene presence/absence matrices, per phylum.
clusters	Named list of character vectors of taxon IDs, per phylum.
method	A function that returns a length-2 numeric vector of effect-size and p-value (see, e.g., phylolm.fx.pv or lm.fx.pv).

`restrict.figfams`
: Optionally, a character vector giving a subset of genes to test.

`drop.zero.var`
: Boolean giving whether to drop genes that are always present or always absent in a particular phylum.

`only.return.names`
: Boolean giving whether to just return the names of genes to be tested (for debugging).

Value

Named list of p-value and effect-size matrices, one per phylum.

---

results.report *Create a summary table giving how many tests were significant.*

Description

Create a summary table giving how many tests were significant.

Usage

```
results.report(results, sigs, signs)
```

Arguments

`results`	List of result matrices, one per phylum.
`sigs`	The output of `make.sigs`.
`signs`	The output of `make.signs`.

Value

A table with the number of positive significant results per phylum at each significance level in `sigs`.

---

retain.observed.taxa *Modify trees to retain only observed taxa (for use with specificity only).*

Description

Modify trees to retain only observed taxa (for use with specificity only).

Usage

```
retain.observed.taxa(trees, phenotype, phenoP, mapped.observed)
```

Arguments

trees	A list of tree objects.
phenotype	A named vector giving the phenotype for each taxon ID.
phenoP	The prior probability of the environment of interest.
mapped.observed	A character vector giving which tips to retain.

Value

An updated list of tree objects.

---

run.burst *Run BURST analysis on a FASTA file of sequences.*

Description

run.burst runs the optimal sequence aligner BURST (doi.org/10.5281/zenodo.806850) on a FASTA file, typically one generated by prepare.burst.input. (In theory, by changing the burst_bin option, any aligner could be used provided it accepts the same command-line options and returns the same formatted output as BURST.)

Usage

```
## S3 method for class 'burst'
run(...)
```

Details

Some particularly relevant global options are:

in_dir String. Path to input directory (i.e., where to look for input files).

burst_infile String. File name of the sequences to be read into BURST.

burst_outfile String. File name where BURST writes output which is then read back into *phylogenize*.

burst_dir String. Path where the binary of BURST is found.

burst_bin String. File name of the binary of BURST.

burst_16sfile String. Path to the 16S FASTA database that maps back to MIDAS species.

data_dir String. Path to directory containing the data files required to perform a *phylogenize* analysis.

Value

Returns TRUE unless an error is thrown.

sanity.check.abundance *Sanity-check abundance data*

Description

sanity.check.abundance is used to make sure that the abundance matrix satisfies the requirements specified by the *phylogenize* application.

Usage

```
sanity.check.abundance(abd.mtx, ...)
```

Arguments

abd.mtx	A matrix or Matrix of abundance or presence values (double or logical).

Value

Always returns TRUE, but will throw errors if the abundance data is the wrong type or class.

sanity.check.metadata *Check that dataset, environment, and sample columns all present*

Description

sanity.check.abundance is used to make sure that the metadata data frame satisfies the requirements specified by the *phylogenize* application.

Usage

```
sanity.check.metadata(metadata, ...)
```

Arguments

metadata	A data frame giving sample annotations.

Details

Some particularly relevant global options are:

env_column Name of metadata column containing environment annotations.

dset_column Name of metadata column containing dataset annotations.

Value

Always returns TRUE, but will throw errors if the metadata does not match specifications.

set_data_internal *Set data directory to internal*

Description

Set data directory to internal

Usage

```
set_data_internal(fail = FALSE, startup = FALSE)
```

Arguments

fail	Boolean. If TRUE, set_data_internal will not attempt to download and install data from Figshare if it is missing.
startup	Boolean. Is this function being called by .onLoad?

single.cluster.plot *Make a hybrid tree-heatmap plot showing the taxon distribution of significant hits.*

Description

Some particularly relevant global options are:

which_phenotype String. Which phenotype to calculate ("prevalence" or "specificity").

gene_color_absent String. When graphing gene presence/absence, this color indicates absence.

gene_color_present String. When graphing gene presence/absence, this color indicates presence.

Usage

```
single.cluster.plot(gene.presence, sig.genes, tree, plotted.tree, phylum,
  verbose = FALSE, ...)
```

Arguments

gene.presence	Gene presence/absence matrix.
sig.genes	Character vector of the significant genes.
tree	A tree object.
plotted.tree	A ggtree plot of tree.
phylum	Name of the phylum represented by tree
verbose	Whether to report debugging information (boolean).

Value

A faceted ggplot object.

sum.nonunique.burst *Sum non-unique rows after BURST mapping.*

Description

sum.nonunique.burst takes BURST results and an abundance or presence/absence matrix, drops any rows that mapped to multiple MIDAS IDs (i.e. that couldn't confidently be assigned to a MIDAS species), then sums any rows that mapped to the same MIDAS ID.

Usage

```
## S3 method for class 'nonunique.burst'
sum(burst, mtx, ...)
```

Arguments

burst	A list obtained by running get.burst.results.
mtx	A presence/absence or abundance matrix, with row names equal to amplicon sequence variant DNA sequences.

Details

Some particularly relevant global options are:

out_dir String. Path to output directory. Default: "output"

in_dir String. Path to input directory (i.e., where to look for input files).

data_dir String. Path to directory containing the data files required to perform a *phylogenize* analysis. Default: on package load, this default is set to the result of system.file("extdata",package="phylogenize").

Value

A new matrix with MIDAS IDs as rows.

tax.annot *Annotate taxa using a taxonomy table.*

Description

Annotate taxa using a taxonomy table.

Usage

```
tax.annot(tns, taxonomy)
```

Arguments

tns	A vector of taxon IDs.
taxonomy	A data frame with at least "cluster" and "species" columns; "cluster" is used to match the identifiers in tns.

Value

A character vector of species names.

---

tbl.result.qvs *Get q-values for results in tbl format.*

Description

Get q-values for results in tbl format.

Usage

```
## S3 method for class 'result.qvs'
tbl(results, method = qvals, ...)
```

Arguments

results	A tbl of results with columns for "phylum" and "p.value".
method	A function: method for obtaining q-values from p-values.
...	Additional parameters to pass to method.

Value

A tbl with an additional "q.value" column.

---

threshold.pos.sigs *Filter out genes that are almost always present or absent.*

Description

Some particularly relevant global options are:

minimum Integer. A particular gene must be observed, and also absent, at least this many times to be reported as a significant positive association with the phenotype.

Usage

```
threshold.pos.sigs(pz.db, phy.with.sigs, pos.sig, ...)
```

Arguments

pz.db	A database for use with *phylogenize* analyses.
phy.with.sigs	A vector of strings giving which phyla had significant results.

Value

A single data frame with entries from results.

Index