Package 'propr'

April 19, 2018

Title Calculating Proportionality Between Vectors of Compositional Data

Version 3.1.9

URL http://github.com/tpq/propr

BugReports http://github.com/tpq/propr/issues

Description The bioinformatic evaluation of gene co-expression often begins with correlation-based analyses. However, this approach lacks statistical validity when applied to relative data. This includes, for example, biological count data generated by high-throughput RNA-sequencing, chromatin immunoprecipitation (ChIP), ChIP-sequencing, Methyl-Capture sequencing, and other techniques. This package implements several metrics for proportionality, including phi [Lovell et al (2015) <DOI:10.1371/journal.pcbi.1004075>] and rho [Erb and Notredame (2016) <DOI:10.1007/s12064-015-0220-8>]. This package also implements several metrics for differential proportionality. Unlike correlation, these measures give the same result for both relative and absolute data.

License GPL-2

LazyData true

VignetteBuilder knitr

RoxygenNote 6.0.1

Depends methods, R (>= 3.2.2)

Imports fastcluster, ggplot2, grDevices, igraph, Rcpp, stats, utils

Suggests ALDEx2, cccrm, compositions, data.table, datasets, directlabels, grid, ggdendro, knitr, limma, plotly, reshape2, rgl, rmarkdown, SDMTools, testthat

LinkingTo Rcpp

NeedsCompilation yes

Author Thomas Quinn [aut, cre],
David Lovell [aut],
Ionas Erb [aut],
Anders Bilgrau [ctb],
Greg Gloor [ctb]

Maintainer Thomas Quinn <contacttomquinn@gmail.com>

Repository CRAN

Date/Publication 2018-04-19 04:10:05 UTC

R topics documented:

`abstract` *Abstract Two propr Objects*

Description

This function abstracts a new `propr` object from two existing `propr` objects. The two `propr` objects should not have any overlapping samples. Typically, the two objects represent different experimental groups. The resultant abstracted object inherits all plot functions available for the original `propr` objects.

Usage

```
abstract(x, y, dt, colBy = "Adjusted", cutoff = 0.01)
```

Arguments

`x, y`	A `propr` object.
`dt`	A `data.table`. The result from `prop2prob(x, y)`.
`colBy`	A character string. The column in `dt` used to select statistically significant pairs.
`cutoff`	A numeric scalar. The value of `colBy` used to select statistically significant pairs.

Details

The abstracted `propr` object has the following properties: The `@counts` and `@logratio` slots contain a join of the original slots via `rbind`. Meanwhile, the `@matrix` slot contains a difference matrix defined as `tanh(atanh(x@matrix) - atanh(y@matrix))`. The `@pairs` slot contains an index of all statistically significant pairs, toggled via the argument `dt`.

Visualizing the difference matrix with `dendrogram` may help summarize the results of `prop2prob`. Note that the difference matrix now also informs co-cluster assignment for the `bucket`, `prism`, and `bokeh` plots. Otherwise, most abstracted plots should match those made using `perb(rbind(x@counts, y@counts))`.

Value

Returns an abstracted `propr` object.

See Also

propr, prop2prob

Examples

```
library(propr)
data(mail)
mail1 <- mail[1:2, ]
mail2 <- mail[3:4, ]
rho1 <- perb(mail1)
rho2 <- perb(mail2)
abstract(rho1, rho2)
```

---

aldex.cor *Correlate CLR with a Continuous Measurement*

Description

This function uses the Monte Carlo instances from an `aldex.clr` object to correlate each log-ratio transformed feature vector with a continuous numeric variable. See `lr2cor`.

Usage

```
aldex.cor(clr, conditions, ...)
```

Arguments

`clr`	An `aldex.clr` object.
`conditions`	A numeric vector of a continuous variable.
`...`	Arguments passed to `cor.test`.

Value

Returns a data.frame of the average correlation statistic (e.g., `r`) and average p-value (`p`) for each feature across all Monte Carlo instances. Average FDR provided by `BH` column.

aldex2propr *Import* ALDEx2 *Object*

Description

This method constructs a propr object from an aldex.clr object. See Details.

Usage

```
aldex2propr(aldex.clr, how = "perb", select)
```

Arguments

aldex.clr	An aldex.clr object.
how	A character string. The proportionality method used to build the propr object. For example, "perb" returns an estimation of rho while "phit" returns an estimation of phi.
select	See proportionality.

Details

The ALDEx2 package has two exceptional features useful in proportionality analysis too. First, ALDEx2 offers a number of extra log-ratio transformations, toggled by the denom argument in aldex.clr. Second, ALDEx2 estimates per-feature technical variation within each sample using Monte-Carlo instances drawn from the Dirichlet distribution.

The aldex2propr function takes advantage of both of these features by constructing a propr object directly from an aldex.clr object. When interpreting the resultant propr object, keep in mind that ALDEx2 adds 0.5 to all @counts regardless of whether the counts contain any zeros. Otherwise, the @logratio slot contains the log-ratio transformed counts as averaged across all Monte Carlo instances. Likewise, the @matrix slot gets filled with the proportionality matrix as averaged across all Monte Carlo instances.

The select argument subsets the feature matrix after log-ratio transformation but before calculating proportionality. This reduces the run-time and RAM overhead without impacting the final result. Removing lowly abundant features prior to log-ratio transformation could otherwise change the proportionality measure.

Value

Returns a propr object.

alphaTheta *Calculate alpha Theta*

Description

Calculate differential proportionality measure, theta, using the Box-Cox transformation method.

Usage

```
alphaTheta(counts, group, alpha)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.
alpha	A double. See vignette for details. Leave missing to skip Box-Cox transformation.

alphaThetaW_old *Calculate Weighted Theta*

Description

Do not use this function. For testing purposes only.

Usage

```
alphaThetaW_old(counts, group, alpha, weights)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.
alpha	A double. See vignette for details. Leave missing to skip Box-Cox transformation.
weights	A matrix. Pre-computed limma-based weights. Optional parameter.

alphaTheta_old *Calculate alpha Theta*

Description

Do not use this function. For testing purposes only.

Usage

```
alphaTheta_old(counts, group, alpha)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.
alpha	A double. See vignette for details. Leave missing to skip Box-Cox transformation.

calculateFDR *Calculate FDR Cutoffs*

Description

This function uses the result of a permuteTheta call to calculate false discovery rate (FDR) cutoffs.

Usage

```
calculateFDR(theta, ptheta, cutoff = seq(0.6, 0.9, 0.1))
```

Arguments

theta	A data.frame. The result of a calculateTheta call.
ptheta	A data.frame. The result of a permuteTheta call.
cutoff	For updateCutoffs, a numeric vector. this argument provides the FDR cutoffs to test for theta. For graph functions, a numeric scalar. This argument indicates the maximum theta to include in the figure. For graph functions, a large integer will instead retrieve the top N pairs as ranked by theta.

calculateFDR_old *Calculate FDR Cutoffs*

Description

Do not use this function. For testing purposes only.

Usage

```
calculateFDR_old(theta, ptheta, cutoff = seq(0.6, 0.9, 0.1))
```

Arguments

theta	A data.frame. The result of a calculateTheta call.
ptheta	A data.frame. The result of a permuteTheta call.
cutoff	For updateCutoffs, a numeric vector. this argument provides the FDR cutoffs to test for theta. For graph functions, a numeric scalar. This argument indicates the maximum theta to include in the figure. For graph functions, a large integer will instead retrieve the top N pairs as ranked by theta.

calculateTheta *Calculate Theta*

Description

Calculate differential proportionality measure, theta. Used by propd to build the @theta slot. A numeric alpha argument will trigger the Box-Cox transformation.

Usage

```
calculateTheta(counts, group, alpha, lrv = NA, only = "all",
  weighted = FALSE, weights = as.matrix(NA))
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.
alpha	A double. See vignette for details. Leave missing to skip Box-Cox transformation.
lrv	A numeric vector. A vector of pre-computed log-ratio variances. Optional parameter.

only	A character string. The name of the theta type to return if only calculating one theta type. Used to make updateCutoffs faster.
weighted	A boolean. Toggles whether to calculate theta using limma::voom weights.
weights	A matrix. Pre-computed limma-based weights. Optional parameter.

Value

A data.frame of theta values.

`calculateThetaW_old` *Calculate Weighted Theta*

Description

Do not use this function. For testing purposes only.

Usage

```
calculateThetaW_old(counts, group)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.

`calculateTheta_old` *Calculate Theta*

Description

Do not use this function. For testing purposes only.

Usage

```
calculateTheta_old(counts, group)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.

caneToad.counts *Example Cane Toad Count Data*

Description

Raw RNA-seq counts for twenty toads sampled from one of two regions in Australia. Data set reduced to exclude any transcripts without at least 10 counts in at least 10 samples.

Usage

```
data(caneToad.counts)
```

Format

An object of class `matrix` with 20 rows and 25774 columns.

Source

<DOI:10.1111/mec.13184>

caneToad.groups *Example Cane Toad Group Data*

Description

Group labels for twenty toads sampled from one of two regions in Australia. Data set reduced to exclude any transcripts without at least 10 counts in at least 10 samples.

Usage

```
data(caneToad.groups)
```

Format

An object of class `character` of length 20.

Source

<DOI:10.1111/mec.13184>

dendroCheck *Dendrogram Plot Check*

Description

Performs data checks before plotting, triggering messages or errors when appropriate. For back-end use only.

Usage

```
dendroCheck()
```

differentialCheck *Differential Proportionality Check*

Description

Performs data checks when comparing two `propr` objects, triggering messages or errors when appropriate. For back-end use only.

Usage

```
differentialCheck(x, y, forceBoth)
```

Arguments

`x`	A `propr` object.
`y`	A `propr` object. Optional.
`forceBoth`	Toggles whether to perform checks for the second `propr` object.

ggdend *Dendrogram Plot Wrapper*

Description

Builds `ggplot2` dendrograms. For back-end use only.

Usage

```
ggdend(dendrogram)
```

Arguments

`dendrogram`	A result from `as.dendrogram`.

Value

Returns a ggplot object.

`ivar2index` *Build Index from ivar Argument*

Description

This function builds an index from the `ivar` argument. Used by the `propr` initialize method and `updateF`.

Usage

```
ivar2index(counts, ivar)
```

Arguments

`counts`	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
`ivar`	A numeric scalar. Specifies reference feature(s) for additive log-ratio transformation. The argument will also accept feature name(s) instead of the index position(s). Set to "iqlr" to use inter-quartile log-ratio transformation. Ignore to use centered log-ratio transformation.

`lr2cor` *Correlate CLR with a Continuous Measurement*

Description

This function correlates each log-ratio transformed feature vector with a continuous numeric variable.

Usage

```
lr2cor(lr, conditions, ...)
```

Arguments

`lr`	A data.frame. A log-ratio transformed counts matrix.
`conditions`	A numeric vector of a continuous variable.
`...`	Arguments passed to `cor.test`.

Value

Returns a data.frame of the correlation statistic (e.g., `r`) and p-value (`p`) for each log-ratio transformed feature. FDR provided by `BH` column.

mail *Mock Mail Count Data*

Description

Includes mock count data for 5 days and 4 zip codes.

Usage

```
data(mail)
```

Format

An object of class `matrix` with 5 rows and 4 columns.

marg.abs *Example Absolute mRNA*

Description

Data generated with supplemental script provided by <DOI:10.1371/journal.pcbi.1004075>. Data originally sourced from <DOI:10.1016/j.cell.2012.09.019>. A time series of yeast mRNA abundance after removal of a key nutrient. Absolute abundance estimated by multiplying microarray signal (relative to first time point) by the initial nCounter-calibrated and copy-per-cell-adjusted RNA-seq abundance (averaged across two replicates). Divide absolute abundances by total sample abundance to make data relative.

Usage

```
data(marg.abs)
```

Format

An object of class `data.frame` with 16 rows and 3031 columns.

migraph igraph *Helper Functions*

Description

igraph Helper Functions

Usage

```
migraph.add(g, names1, names2, force = TRUE)

migraph.color(g, names1, names2, col)

migraph.clean(g)
```

Arguments

g	An igraph object.
names1, names2	A character vector. The names1 argument defines a first set of vertices. The names2 argument defines a second set of vertices to which the first set connects (i.e., element-wise), thereby defining a set of edges.
force	A boolean. If true, the function adds any missing vertices before adding edges. If false, the function only adds edges that have both vertices already present.
col	A character string. The color applied to all vertices (or edges) specified by the names1 (or names2) argument.

Value

An igraph object.

multiplot *Plot Multiple Graphs*

Description

Easily plot multiple graphs within the same window. Code adapted from http://www.cookbook-r.com/. For back-end use only.

Usage

```
multiplot(..., cols = 1)
```

Arguments

...	Multiple plots.
cols	A numeric scalar. The number of plot columns.

packageCheck *Package Check*

Description

Checks whether the user has the required package installed. For back-end use only.

Usage

```
packageCheck(package)
```

Arguments

package	A character string. An R package.

pals *Calculate PALs for Pairs*

Description

This function finds the Popular Adjacent Ligand or Self (PALs) for each feature in the `@theta` slot of a `propd` object. Specifically, we define PALs as the adjacent node with the highest amount of connectivity. If node itself has more connectivity than any of its neighbors, it is its own PAL.

Usage

```
pals(object, k)
```

Arguments

object	A `propd` object.
k	An integer. The maximum number of PALs to index when calculating `pals` in the network.

Value

A named vector of PALs, ordered by decreasing connectivity of the input nodes. The names of the result refer to the input nodes themselves.

pd.d *Example propd Object*

Description

Includes results from propd as applied to cane toad transcripts with at least 40 counts in at least 20 samples (after removing any transcripts with 0 counts). The resultant object is filtered to include only the top 1000 theta_d values in the @theta slot.

Usage

```
data(pd.d)
```

Format

An object of class propd of length 1.

Source

<DOI:10.1111/mec.13184>

pd.e *Example propd Object*

Description

Includes results from propd as applied to cane toad transcripts with at least 40 counts in at least 20 samples (after removing any transcripts with 0 counts). The resultant object is filtered to include only the top 1000 theta_e values in the @theta slot.

Usage

```
data(pd.e)
```

Format

An object of class propd of length 1.

Source

<DOI:10.1111/mec.13184>

permuteTheta *Permute Theta*

Description

Permute differential proportionality measure, theta.

Usage

```
permuteTheta(counts, group, p = 64)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.
p	An integer. The number of permutation cycles.

Details

This function randomizes group membership p*nfeat times.

permuteTheta_false *Permute Theta*

Description

Permute differential proportionality measure, theta.

Usage

```
permuteTheta_false(counts, group, p = 64)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.
p	An integer. The number of permutation cycles.

Details

For back-end use only.

permuteTheta_naive *Permute Theta*

Description

Permute differential proportionality measure, theta.

Usage

```
permuteTheta_naive(counts, group, p = 64)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.
p	An integer. The number of permutation cycles.

Details

This function randomizes all feature vectors p times.

permuteTheta_old *Permute Theta*

Description

Do not use this function. For testing purposes only.

Usage

```
permuteTheta_old(counts, group, p)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.
p	An integer. The number of permutation cycles.

permuteTheta_prime *Permute Theta*

Description

Permute differential proportionality measure, theta.

Usage

```
permuteTheta_prime(counts, group, p = 64)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
group	A character vector. Group or sub-group memberships, ordered according to the row names in counts.
p	An integer. The number of permutation cycles.

Details

This function randomizes group labels p times.

plotCheck *Plot Check*

Description

Performs data checks before plotting, triggering messages or errors when appropriate. For back-end use only.

Usage

```
plotCheck(rho, prompt, plotly, indexNaive)
```

Arguments

rho	A propr object created from perb. However, smear, dendrogram, and cytescape will also accommodate results from phit and phis.
prompt	A logical scalar. Set to FALSE to disable the courtesy prompt when working with big data.
plotly	A logical scalar. Set to TRUE to produce a dynamic plot using the plotly package.
indexNaive	Toggles whether to perform checks for an "index-naive" plot function.

Value

Returns a propr object with guaranteed column names and row names.

---

progress *Make Progress Bar*

Description

Make Progress Bar

Usage

```
progress(i, k, numTicks)
```

Arguments

i	The current iteration.
k	Total iterations.
numTicks	The result of progress.

Value

The next numTicks argument.

---

promptCheck *Feature Check*

Description

Prompts user when performing an operation on an unusually large set of features. For back-end use only.

Usage

```
promptCheck(N)
```

Arguments

N	An integer. The number of features.

prop2prob *Calculate Probability from Proportionality*

Description

This experimental helper function calculates probability from proportionality. When supplying one propr object, prop2prob estimates the probability that each proportionality coefficient occurred by chance alone. When supplying two propr objects, prop2prob estimates the probability that each proportionality coefficient differs between the two objects.

Usage

```
prop2prob(x, y, method = "bonferroni", prompt = TRUE)
```

Arguments

x	A propr object.
y	A propr object. Optional.
method	A character string. Selects method used to adjust p-values for multiple comparisons. Argument passed to p.adjust. Defaults to the more conservative Bonferroni correction.
prompt	A logical scalar. Set to FALSE to disable the courtesy prompt when working with big data.

Details

All calculations use formulae derived for the concordance correlation coefficient under the constraint that all means equal zero. We defend this constraint on the grounds that we can shift the mean of log-ratio transformed feature vectors without changing the proportionality coefficient, rho, or Pearson's correlation coefficient, r. We refer the reader to Zar's Biostatistical Analysis text (4ed, pg 407-10) for more information on the method used.

When calculating differential proportionality, it is the responsibility of the user to ensure that the two groups have no overlapping samples. All p-values returned as twice the result of pnorm, thereby correcting for "two-tails". Please make sure to interpret p-values in the context of multiple testing! For more information, see p.adjust.

Value

Returns a data.table of p-values.

See Also

propr, abstract

Examples

```
library(propr)
data(mail)
rho <- perb(mail)
prop2prob(rho)
```

---

propd *The propd Method*

Description

Welcome to the propd method!

Let X and Y be non-zero positive feature vectors measured across N samples belonging to one of two groups, sized $N1$ and $N2$. We use VLR to denote the variance of the log of the ratio of the vectors X over Y. We define theta as the weighted sum of the within-group VLR divided by the weighted total VLR.

The propd method calculates theta. However, VLR fails in the setting of zero counts. The propd method will use a Box-Cox transformation to approximate VLR based on the parameter α if provided. We refer the user to the vignette for more details.

Note that Group 1 always refers to the first element of the group vector argument supplied to propd.

Usage

```
## S4 method for signature 'propd'
show(object)

propd(counts, group, alpha, p = 100, weighted = FALSE)

propd2propr(object, ivar)

setActive(propd, what = "theta_d")

setDisjointed(propd)

setEmergent(propd)

## S4 method for signature 'propd,missing'
plot(x, y, cutoff = 1000, col1, col2, propr,
  prompt = TRUE, d3 = FALSE, plotSkip = FALSE)

shale(object, cutoff = 1000, k, prompt = TRUE, clean = FALSE)

geyser(object, cutoff = 1000, k = 5, prompt = TRUE, plotly = FALSE)

bowtie(object, cutoff = 1000, k = 5, prompt = TRUE, plotly = FALSE)
```

```
gemini(object, cutoff = 1000, k = 5, prompt = TRUE, plotly = FALSE)

slice(object, cutoff = 1000, reference, prompt = TRUE, plotly = FALSE)

decomposed(object, cutoff = 1000, prompt = TRUE)

updateCutoffs(propd, cutoff = seq(0.05, 0.95, 0.3))

updateF(propd, moderated = FALSE, ivar = "clr")
```

Arguments

`object, x, propd`	A propd object.
`counts`	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
`group`	A character vector. Group or sub-group memberships, ordered according to the row names in `counts`.
`alpha`	A double. See vignette for details. Leave missing to skip Box-Cox transformation.
`p`	An integer. The number of permutation cycles.
`weighted`	A boolean. Toggles whether to calculate theta using `limma::voom` weights.
`ivar`	A numeric scalar. Specifies reference feature(s) for additive log-ratio transformation. The argument will also accept feature name(s) instead of the index position(s). Set to "iqlr" to use inter-quartile log-ratio transformation. Ignore to use centered log-ratio transformation.
`what`	A character string. The theta type to set active.
`y`	Missing. Ignore. Leftover from the generic method definition.
`cutoff`	For `updateCutoffs`, a numeric vector. this argument provides the FDR cutoffs to test for theta. For graph functions, a numeric scalar. This argument indicates the maximum theta to include in the figure. For graph functions, a large integer will instead retrieve the top N pairs as ranked by theta.
`col1`	A character vector. Specifies which nodes to color `red` or `blue`, respectively.
`col2`	A character vector. Specifies which nodes to color `red` or `blue`, respectively.
`propr`	An indexed `propr` object. Use to add proportional edges (colored green) to a `propd` network.
`prompt`	A logical scalar. Set to `FALSE` to disable the courtesy prompt when working with big data.
`d3`	A boolean. Use `rgl` to plot 3D network.
`plotSkip`	A boolean. Toggles whether to build the network graph without plotting it. Used by `pals`.
`k`	An integer. The maximum number of PALs to index when calculating `pals` in the network.

clean	A boolean. Toggles whether to remove pairs with "Bridged" or "Missing" PALs. Used by geyser, bowtie, and gemini.
plotly	A logical scalar. Set to TRUE to produce a dynamic plot using the plotly package.
reference	A character string. A feature to use as the denominator reference when comparing log-ratio abundances.
moderated	For updateF, a boolean. Toggles whether to calculate a moderated F-statistic.

Details

setActive: Build analyses and figures using a specific theta type. For example, set what = "theta_d" to analyze disjointed proportionality and what = "theta_e" to analyze emergent proportionality. These provide the same results as setDisjointed and setEmergent, respectively.

updateCutoffs: Use the propd object to permute theta across a number of theta cutoffs. Since the permutations get saved when the object is created, calling updateCutoffs will use the same random seed each time.

updateF: Use the propd object to calculate the F-statistic from theta as described in the Erb et al. 2017 manuscript on differential proportionality. Optionally calculates a moderated F-statistic using the limma-voom method. Supports weighted and alpha transformed theta values.

plot: Plots the interactions between pairs as a network. When plotting disjointed proportionality, red edges indicate that LRM1 > LRM2 while blue edges indicate that LRM1 < LRM2. When plotting emergent proportionality, red edges indicate that VLR1 < VLR2 while blue edges indicate that VLR1 > VLR2. Group labels numbered based on the order of the group argument to propd. Use col1 and col2 arguments to color nodes. For more control over the visualization of the network, consider exporting the table from shale to a network visualization tool like Cytoscape.

shale: Builds a table of within-group and total log-ratio variances, log-ratio means, and PALs (see: pals). If the argument k is provided, the table will label at most k top PALs. Just as each node gets assigned a PAL, shale aims to assign each edge a PAL. Edges that have a top PAL as one and only one of their nodes get assigned that PAL. Edges that have top PALs as both of their nodes get assigned "Bridged". Edges without a top PAL as one of their nodes will get assigned a PAL if either (a) both nodes have the same neighbor PAL or (b) one node has a "Missing" neighbor PAL. The cutoff argument guides the maximum value of theta above which to exclude the pair. A large integer cutoff will instead retrieve the top N pairs as ranked by theta.

geyser: Plots indexed pairs based on the within-group log-ratio variance (VLR) for each group. Pairs near the origin show a highly proportional relationship in both groups. Each line away from the y = x line indicates a doubling of VLR compared to the other group. All pairs colored based on PAL (see: pals). See gemini.

bowtie: Plots indexed pairs based on the log-ratio means (LRM), relative to its PAL, for each group. Pairs near the origin show comparable LRM, relative to its PAL, in both groups. Each line away from the y = x line indicates a doubling of LRM compared to the other group. All pairs colored based on PAL (see: pals). See gemini.

gemini: Plots indexed pairs based on the log-fold difference in log-ratio variance (VLR) between the two groups versus the difference in log-ratio means (LRM). In this figure, the LRM for each group is signed (i.e., positive or negative) such that the PAL is the denominator of the log-ratio. This allows for a fluid comparison between pairs within the same PAL module. Pairs with a "Bridged"

or "Missing" PAL get excluded from this graph. Remember that an increase in VLR suggests less proportionality. All pairs colored based on PAL (see: pals).

slice: Plots log-ratio counts relative to a reference node for all pairs that include the reference itself. This makes a useful adjunct function to visualize how features vary across samples relative to a PAL.

decomposed: Plots the decomposition of log-ratio variance into (weighted) group variances and between-group variance. Useful for visualizing how a theta type selects pairs.

propd2propr: Transforms a propd object into a propr object where the @matrix slot contains $1 - \theta$. Allows the user to interrogate theta using any function described in visualize.

Slots

counts A data.frame. Stores the original "count matrix" input.

group A character vector. Stores the original group labels.

alpha A double. Stores the alpha value used for transformation.

weighted A logical. Stores whether the theta is weighted.

weights A matrix. If weighted, stores the limma-based weights.

active A character. Stores the name of the active theta type.

theta A data.frame. Stores the pairwise theta measurements.

Fivar ANY. Stores the reference used to moderate theta.

dfz A double. Stores the prior df used to moderate theta.

permutes A data.frame. Stores the shuffled group labels, used to reproduce permutations of theta.

fdr A data.frame. Stores the FDR cutoffs for theta.

Methods (by generic)

show: Method to show propd object.

plot: Method to plot propd object.

---

proportionality *Calculate Proportionality*

Description

Let D represent any number of features measured across N biological replicates exposed to a binary or continuous event E. For example, E could indicate case-control status, treatment status, treatment dose, or time. This function converts a "count matrix" with N rows and D columns into a proportionality matrix of D rows and D columns.

For phi, the result of phit, one can think of the resultant matrix as analogous to a distance matrix, except that it has no symmetry unless forced. For rho, the result of perb, one can think of the resultant matrix as analogous to a correlation matrix. For phs, the result of phis, one can think of the resultant matrix as either a naturally symmetric variant of phi or a monotonic variant of rho. Another function, corr, calculates Pearsons' correlation using log-ratio transformed data.

These methods all use the centered log-ratio transformation by default, but will use an additive log-ratio transformation instead if a scalar `ivar` is provided. When using an additive log-ratio transformation, this function will return `rho = 0` for the column and row in the `@matrix` slot that would contain the reference feature. Setting `ivar` to a numeric or character vector will transform data using the geometric mean of only the indexed features. Alternatively, setting `ivar` to "iqlr" will transform data using the geometric mean of only the features with variances that fall in the inter-quartile range of all per-feature variances. We base this "iqlr" transformation on the `ALDEx2` package.

Log-ratio transformation, by its nature, fails if the input data contain any zero values. To avoid an error in this case, these methods automatically replace all zero values with 1. However, the topic of zero replacement is controversial. Proceed carefully when analyzing data that contain any zero values.

The `select` argument subsets the feature matrix after log-ratio transformation but before calculating proportionality. This reduces the run-time and RAM overhead without impacting the final result. Removing lowly abundant features prior to log-ratio transformation could otherwise change the proportionality measure.

Usage

```
## S4 method for signature 'propr'
initialize(.Object, counts, ivar, select)

phit(counts, ivar = 0, select, symmetrize = TRUE)

perb(counts, ivar = 0, select)

phis(counts, ivar = 0, select)

corr(counts, ivar = 0, select)
```

Arguments

`.Object`	Missing. Ignore. Leftover from the generic method definition.
`counts`	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
`ivar`	A numeric scalar. Specifies reference feature(s) for additive log-ratio transformation. The argument will also accept feature name(s) instead of the index position(s). Set to "iqlr" to use inter-quartile log-ratio transformation. Ignore to use centered log-ratio transformation.
`select`	Subsets via `object@counts[, select]`. Optional. Use this argument to subset the proportionality matrix before building without altering the final result.
`symmetrize`	A logical. If `TRUE`, forces symmetry by reflecting the "lower left triangle".

Value

Returns a `propr` object.

Examples

```
library(propr)
data(mail)
phi <- phit(mail)
rho <- perb(mail)
phs <- phis(mail)
```

---

propr *The propr Package*

Description

Welcome to the propr package!

To learn more about calculating proportionality, see proportionality.

To learn more about visualizing proportionality, see visualize.

To learn more about ALDEx2 package integration, see aldex2propr.

To learn more about differential proportionality, see propd.

To learn more about compositional data analysis, and its relevance to biological count data, see the bundled vignette.

Usage

```
## S4 method for signature 'propr'
show(object)

## S4 method for signature 'propr'
subset(x, subset, select)

## S4 method for signature 'propr,ANY,ANY'
x[i = "all", j, tiny = FALSE]

## S4 method for signature 'propr,missing'
plot(x, y, prompt = TRUE, plotly = FALSE)

simplify(object)

adjacent(object)
```

Arguments

object, x	An object of class propr.
subset	Subsets via object@counts[subset,]. Use this argument to rearrange subject order.
select	Subsets via object@counts[, select]. Use this argument to rearrange feature order.

i	Operation used for the subset indexing. Select from "==", "=", ">", ">=", "<", "<=", "!=", or "all".
j	Reference used for the subset indexing. Provide a numeric value to which to compare the proportionality measures in the @matrix slot.
tiny	A logical scalar. Toggles whether to pass the indexed result through simplify.
y	Missing. Ignore. Leftover from the generic method definition.
prompt	A logical scalar. Set to FALSE to disable the courtesy prompt when working with big data.
plotly	A logical scalar. Set to TRUE to produce a dynamic plot using the plotly package.

Slots

counts A matrix. Stores the original "count matrix" input.

logratio A matrix. Stores the log-ratio transformed "count matrix".

matrix A matrix. Stores the proportionality matrix calculated by phiRcpp or rhoRcpp.

pairs A vector. Indexes the proportionality metrics of interest.

Methods (by generic)

show: Method to show propr object.

subset: Method to subset propr object.

[: Method to subset propr object.

plot: Method to plot propr object.

Functions

simplify: This convenience function takes an indexed propr object and subsets the object based on that index. Then, it populates the @pairs slot of the new object with an updated version of the original index. You can call simplify from within the [method using the argument tiny.

adjacent: This function uses pairs indexed in the @pairs slot to build a symmetric adjacency matrix.

---

proprALR *Calculates the additive log-ratio transformation.*

Description

Provided for backend use.

Usage

```
proprALR(X, ivar, check = FALSE)
```

Arguments

X	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
ivar	A numeric scalar. Specifies feature to use as reference for additive log-ratio transformation.
check	A logical. If TRUE, function first checks for negative and NA values.

Value

A matrix. Returns the additive log-ratio transformation of X.

---

proprCLR *Calculates the centered log-ratio transformation.*

Description

Provided for backend use.

Usage

```
proprCLR(X, check = FALSE)
```

Arguments

X	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
check	A logical. If TRUE, function first checks for negative and NA values.

Value

A matrix. Returns the centered log-ratio transformation of X.

---

proprPairs *Recasts proportionality matrix as a table of feature pairs.*

Description

Provided for backend use.

Usage

```
proprPairs(prop)
```

Arguments

prop	A data.frame or matrix. A proportionality matrix.

Value

A data.frame. Returns a table of feature pairs.

---

proprPerb *Calculate proportionality metric rho (Erb 2016).*

Description

Provided for backend use.

Usage

```
proprPerb(counts, ivar = 0)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
ivar	A numeric scalar. Specifies reference feature(s) for additive log-ratio transformation. The argument will also accept feature name(s) instead of the index position(s). Set to "iqlr" to use inter-quartile log-ratio transformation. Ignore to use centered log-ratio transformation.

Value

Returns proportionality matrix.

---

proprPhit *Calculate proportionality metric phi (Lovell 2015).*

Description

Provided for backend use.

Usage

```
proprPhit(counts, symmetrize = TRUE)
```

Arguments

counts	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
symmetrize	A logical. If TRUE, forces symmetry by reflecting the "lower left triangle".

Value

Returns proportionality matrix.

---

proprSym *Symmetrizes a proportionality matrix.*

Description

Provided for backend use.

Usage

```
proprSym(prop)
```

Arguments

prop	A data.frame or matrix. A proportionality matrix.

Value

A matrix. Returns a symmetrized proportionality matrix.

---

proprTri *Retrieve the lower left triangle of a proportionality matrix.*

Description

Provided for backend use.

Usage

```
proprTri(prop)
```

Arguments

prop	A data.frame or matrix. A proportionality matrix.

Value

A vector. Returns the lower left triangle of a proportionality matrix.

proprVLR *Calculates the variance of the log of the ratios.*

Description

Provided for backend use.

Usage

```
proprVLR(X, check = FALSE)
```

Arguments

X	A data.frame or matrix. A "count matrix" with subjects as rows and features as columns.
check	A logical. If TRUE, function first checks for negative and NA values.

Value

Returns a matrix containing the variance of the log of the ratios.

ratios *Recast Matrix as Feature Ratios*

Description

The `ratios` function recasts a matrix with N feature columns as a new matrix with N * (N - 1) / 2 feature ratio columns.

Usage

```
ratios(matrix)
```

Arguments

matrix	A matrix. The data to recast.

Value

A matrix.

top.counts *Example propr Object*

Description

Includes the non-transformed count abundances from all cane toad transcripts with at least 10 counts in at least 10 samples, subsetted to include only those indexed by `rho > .995`. Used for vignette.

Usage

```
data(top.counts)
```

Format

An object of class `matrix` with 20 rows and 409 columns.

Source

<DOI:10.1111/mec.13184>

top.lr *Example propr Object*

Description

Includes the log-ratio transformed abundances from all cane toad transcripts with at least 10 counts in at least 10 samples, subsetted to include only those indexed by `rho > .995`. Used for vignette.

Usage

```
data(top.lr)
```

Format

An object of class `matrix` with 20 rows and 409 columns.

Source

<DOI:10.1111/mec.13184>

visualize *Visualize Proportionality*

Description

smear: Plots log-ratio transformed abundances pairwise. Index-aware, meaning that it only plots pairs indexed in @pairs, unless no pairs are indexed.

dendrogram: Plots a clustering of proportionality matrix. Index-aware, meaning that it only plots pairs indexed in @pairs, unless no pairs are indexed. Heatmap intensity is not scaled.

slate: Builds a table of VLR, VLS, and proportionality for each feature pair in a propr object. If the argument k is provided, the table will also include co-cluster membership.

prism: Plots the variance of the ratio of the log-ratio transformed feature pair (VLR) versus the sum of the individual variances of each log-ratio transformed feature (VLS). The ratio of the VLR to the VLS equals 1 - rho. As such, we use here seven rainbow colored lines to indicate where rho equals [.01, .05, .50, 0, 1.50, 1.95, 1.99], going from red to violet.

bokeh: Plots the feature variances for each log-ratio transformed feature pair in the propr object. Highly proportional pairs will aggregate near the y = x diagonal. Clusters that appear toward the top-right of the figure contain features with highly variable abundance across all samples. Clusters that appear toward the bottom-left of the figure contain features with fixed abundance across all samples. Uses a log scale.

bucket: Plots an estimation of the degree to which a feature pair differentiates the experimental groups versus the measure of the proportionality between that feature pair. The discrimination score is defined as the negative log of the p-values for each feature in the pair, computed independently using kruskal.test. "It's pronounced, 'bouquet'." - Hyacinth Bucket

pca: Plots the first two principal components as calculated using the log-ratio transformed feature vectors. This provides a statistically valid alternative to conventional principal components analysis (PCA). For more information, see <DOI:10.1139/cjm-2015-0821>.

snapshot: Plots the log-ratio transformed feature abundance as a heatmap, along with the respective dendrograms. Heatmap intensity is not scaled.

cytescape: Builds a table of indexed pairs and their proportionality. In doing so, this function displays a preview of the interaction network, built using igraph. We recommend using the result as input to a network visualization tool like Cytoscape.

Usage

```
smear(rho, prompt = TRUE, plotly = FALSE)

dendrogram(rho, prompt = TRUE, plotly = FALSE)

slate(rho, k, prompt = TRUE, plotly = FALSE)

bucket(rho, group, k, prompt = TRUE, plotly = FALSE)

prism(rho, k, prompt = TRUE, plotly = FALSE)
```

```
bokeh(rho, k, prompt = TRUE, plotly = FALSE)

pca(rho, group, prompt = TRUE, plotly = FALSE)

snapshot(rho, prompt = TRUE, plotly = FALSE)

cytescape(object, col1, col2, prompt = TRUE, d3 = FALSE)
```

Arguments

prompt	A logical scalar. Set to FALSE to disable the courtesy prompt when working with big data.
plotly	A logical scalar. Set to TRUE to produce a dynamic plot using the plotly package.
k	A numeric scalar. The number of clusters. Optional parameter for bucket, prism, and bokeh. Providing the argument k will color feature pairs by co-cluster membership. In other words, a feature pair will receive a color if and only if both features belong to same the cluster (calculated using hclust).
group	A character vector. Group or sub-group memberships, ordered according to the row names in @counts and @logratio. Required parameter for bucket and optional parameter for pca.
object, rho	A propr object created from perb. However, smear, dendrogram, and cytescape will also accommodate results from phit and phis.
col1, col2	A character vector. Specifies which nodes to color red or blue, respectively.
d3	A boolean. Use rgl to plot 3D network.

Value

smear, pca: Returns a ggplot object.

dendrogram, snapshot: Returns a dendrogram object.

slate: Returns a data.frame of all pairwise relationships. If the argument k is provided, returns a list of the data.frame of pairwise relationships and the cluster membership.

prism, bokeh, bucket: Returns cluster membership if k is provided. Otherwise, returns a ggplot object.

cytescape: Returns a data.frame of indexed pairs.

Index