Package ‘eudysbiome’

April 23, 2016

Type Package

Title pseudo-cartesian plot and contingency test on 16S Microbial data

Version 1.0.0

Date 2015-04-07

Author Xiaoyuan Zhou, Christine Nardini

Maintainer Xiaoyuan Zhou `<zhouxiaoyuan@picb.ac.cn>`

Description eudysbiome a package that permits to annotate the differential genera as harmful/harmless based on their ability to contribute to host diseases (as indicated in literature) or as unknown based on their ambiguous genus classification. Further, the package statistically measures the eubiotic (harmless genera increase or harmful genera decrease) or dysbiotic(harmless genera decrease or harmful genera increase) impact of a given treatment or environmental change on the (gut-intestinal, GI) microbiome in comparison to the microbiome of the reference condition.

Depends R (>= 3.2.1)

Imports plyr(>= 1.8.1)

biocViews Microbiome, Metagenomics,DifferentialExpression, Annotation, Visualization, MultipleComparison, SystemsBiology, Classification, Sequencing, Software

License GPL-2

LazyData true

NeedsCompilation no

R topics documented:

`contingencyCount` *Contingency Table Construction*

Description

Computes the frequencies of the contingency table as the accumulated microbial abundance difference classified into each condition and eubiotic/dysbiotic impact term for examining the significance of the association (contingency) between conditions and impacts by `contingencyTest`.

Usage

```
contingencyCount(x, micro.anno=NULL, comp.anno=NULL)
```

Arguments

`x`	See x in `pseudoCartesian`, the x values should be difference values without log converted.
`micro.anno`	See `micro.anno` in `pseudoCartesian`.
`comp.anno`	See `comp.anno` in `pseudoCartesian`.

Details

Eubiotic impact is measured by variations of increased harmless and decreased harmful microbes, while the dysbiotic impact is measured by the decreased harmless and increased harmful microbes.

Value

The frequencies of condition-impact terms in contingency table

Examples

```
data(microDiff)
attach(microDiff)

microCount = contingencyCount(x=data, micro.anno=micro.anno,
                      comp.anno = comp.anno)

detach(microDiff)
```

`contingencyTest` *Contingency test for count data*

Description

Performs Chi-squared test or Fisher's exact test for testing the significance of association between conditions and eubiotic/dysbiotic impacts in a contingency table.

Usage

```
contingencyTest(microCount, chisq = TRUE, fisher =TRUE,
                alternative=c("greater"))
```

Arguments

`microCount`	a *m by 2* data frame or numeric matrix of contingency table with frequencies under each condition-impact term; could be produced from `contingencyCount`.
`chisq,fisher`	logical indicating if the Chi-squared test or Fisher's exact test should be performed.
`alternative`	parameter specifying for alternative hypothesis, only used when `fisher` is `TRUE`; see `fisher.test`.

Details

Chi-squared test for testing the probability that the proportions of eubiotic frequencies are different between two conditions; furtherly, the Fisher's exact test for testing whether one condition is more likely to be associated to eubiotic impact. More details, refer to `chisq.test` and `fisher.test`

Value

A list with following components: Chisq Chi-squared test results for each pair-wise condition. Chisq.p the p-values of the Chi-squared tests for all pair-wise conditions. Fisher Fisher's exact test results for each pair-wise condition. Fisher.p the p-values of the Fisher's exact tests for all pair-wise conditions.

See Also

`contingencyCount`, `fisher.test`, `chisq.test`

Examples

```
data(microCount)

contingencyTest(microCount,chisq=TRUE,fisher =TRUE,
          alternative ="greater")
```

`diffGenera` *Differential microbes in Genus-Species table A data frame containing 10 differential genera and the species included, which was to be annotated as* `"harmful"` *or* `"harmless"`*.*

Description

Differential microbes in Genus-Species table A data frame containing 10 differential genera and the species included, which was to be annotated as `"harmful"` or `"harmless"`.

Usage

```
data(diffGenera)
```

Format

A data frame with 26 rows and 2 columns specifying for `Genus` and `Species`.

Value

A data frame of 10 differential genera and species included.

`eudysbiome` *eudysbiome.*

Description

eudysbiome.

`harmGenera` *Manually curated genera annotation table A data frame containing 59 genera annotated as "harmful" and the harmful species included in these genera.*

Description

Manually curated genera annotation table A data frame containing 59 genera annotated as "harmful" and the harmful species included in these genera.

Usage

```
data(harmGenera)
```

Format

A data frame with 118 rows and 2 columns specifying for `Genus` and `Species`.

Value

A data frame of 59 harmful genera and species included.

`microAnnotate` *Genus Annotation*

Description

Annotates given genera as harmful or harmless based on either our manually curated, harmful Genus-Species table in data `harmGenera` of this package or user defined table.

Usage

```
microAnnotate(microbe,species=TRUE, annotated.micro=NULL)
```

Arguments

`microbe`	a genus list to be annotated; a Genus-Species data frame which represents the genera and the included corresponding species is recommended to be provided by users for the more accurate annotations.
`species`	logical, specifying if the species are provided in the `microbe` for the annotations; default to `TRUE`.
`annotated.micro`	the annotated genera which are used for the annotation of `microbe`, it could either be loaded from the data `harmGenera` or defined by users.

Value

The annotated genera.

Examples

```
#load the genera to be annotated
data(diffGenera)

#load the curated Genus-Species annotation table
data(harmGenera)

microAnnotate(microbe = diffGenera, species =TRUE,
          annotated.micro= harmGenera)
```

`microCount` *Microbial count contingency table*

Description

A matrix containing the counts of differential microbe classified into each condition-eubiotic/dysbiotic impact couple. Rows represent the condition comparisons, columns represent the eubiotic and dysbiotic impacts:

Usage

```
data(microCount)
```

Format

A data frame with 2 rows and 2 variables

Details

- EI. eubiotic impact
- DI. dysbiotic impact

The table can be produced by `microCount` function.

Value

A data frame of differential abundance counts in condition-impact couple.

`microDiff` *Differential annotated genera with abundance variations among pairwise condition comparisons*

Description

A list containing: i) a data frame of 10 differentila genera with abundance differences among 3 condition comparisons, in which row represents the differential microbes and column represents the comparisons; ii) Genera annotations for the 10 differential genera; iii) pre-defined condition comparison names

Usage

```
data(microDiff)
```

Format

A list

Value

A list of differential genera and annotations

`pseudoCartesian` *Pseudo-2-D Cartesian Plane Plots*

Description

This function plots a Cartesian plane of genus abundance difference across the tested conditions (y-axis) and their harmful/harmless nature (negative/positive x-axis), giving rise to up-utmost right and bottom-utmost left quadrants of microbial eubiotic impact and bottom-utmost right and up-utmost left quadrants of dysbiotic impact.

Usage

```
pseudoCartesian(x,log2 =TRUE,micro.anno=NULL, comp.anno=NULL,
             unknown=FALSE, pch=16, point.col= NULL,point.alpha =0.6,
             ylim=NULL, xlab=NULL,ylab=NULL,vlty= 2, hlty = 1, srt =60,
             font=3, adj=c(1,1), xaxis=NULL, yaxis=NULL, legend = TRUE,
             shade=TRUE,shade.col =c("darkgreen","red"),shade.alpha=0.2,
             ...)
```

Arguments

`x`	a data frame or numeric matrix of microbial abundance variations from which the plot is produced. Rows indicate the differential microbes, columns indicate the pair-wise conditions. `x` values can either be difference values or be *log2* converted, specified with `log2` parameter.
`log2`	logical, specifying if x values should be *log2* converted; default to `TRUE`.
`micro.anno`	a character vector to annotate all row microbes in `x`; e.g. `"harmless"`,`"harmful"`, should be in same length with the microbes. It can be given by the output of `microAnnotate`
`comp.anno`	a character vector of conditions pre-defined from the column pair-wise comparisons, should be in same length with the comparisons; default to the pair-wise comparisons.
`unknown`	logical, specifying if unknown microbes are shown in the planes; default to `FALSE`.
`pch`	a vector of point types. Graphical parameters:see `par`.
`point.col`	a vector of colors for the points.
`point.alpha`	alpha value for points; see `adjustcolor`.
`ylim`	limits for the y axis.
`xlab`	a title for the x axis.
`ylab`	a title for the y axis.

`vlty,hlty`	types of vertical and horizontal lines to divide the plane with x-axis and y-axis, respectively.
`srt,font,adj`	graphical parameters for the text on x-axis, see `par`.
`xaxis`	a character or expression vector specifying the labels of x axis by *text*; default to row names of `x`.
`yaxis`	a character or expression vector specifying the labels of y axis by *axis*; default to `at` values in `axis`.
`legend`	logical, specifying if the legend should be added to the plot; default to `TRUE`.
`shade`	logical, specifying if the quadrants should be highlighted by shades; default to `TRUE`.
`shade.col`	a vector of colors for the up-utmost right, bottom-utmost left quadrants and bottom-utmost right, up-utmost left quadrants, respectively; default to "darkgreen" and "red". If only one color is specified, the other one can be `NA`.
`shade.alpha`	alpha value for shades; see `adjustcolor`.
`...`	additional parameters passed to the default method, or by it to `plot.window`, `text`,`mtext`,`axis`, and `title` to control the appearance of the plot.

Value

The pseudo-Cartesian plane plot

Examples

```
data(microDiff)
attach(microDiff)

newpar = par()
par(mar = c(6,5.1,4.1,6))
pseudoCartesian(x=data,log2=TRUE,micro.anno = micro.anno,pch =16,
         comp.anno= comp.anno,unknown=TRUE,
         point.col=c(”blue”,”purple”,”orange”))

par(newpar)
detach(microDiff)
```

Index