Package: Anaconda (via r-universe)

November 5, 2024

Type Package

Title Targeted Differential and Global Enrichment Analysis of Taxonomic Rank by Shared Asvs

Version 0.1.5

Author Pierre-Louis Stenger [cre, aut]

Maintainer Pierre-Louis Stenger <`Pierrelouis.stenger@gmail.com`>

Description Targeted differential and global enrichment analysis of taxonomic rank by shared ASVs (Amplicon Sequence Variant), for high-throughput eDNA sequencing of fungi, bacteria, and metazoan. Actually works in two steps: I) Targeted differential analysis from QIIME2 data and II) Global analysis by Taxon Mann-Whitney U test analysis from targeted analysis (I) (I) Estimate variance-mean dependence in count/abundance ASVs data from high-throughput sequencing assays and test for differential represented ASVs based on a model using the negative binomial distribution. (II) NCBITaxon_MWU uses continuous measure of significance (such as fold-change or -log(p-value)) to identify NCBITaxon that are significantly enriches with either up- or down-represented ASVs. If the measure is binary (0 or 1) the script will perform a typical 'NCBITaxon enrichment' analysis based Fisher's exact test: it will show NCBITaxon over-represented among the ASVs that have 1 as their measure. On the plot, different fonts are used to indicate significance and color indicates enrichment with either up (red) or down (blue) regulated ASVs. No colors are shown for binary measure analysis. The tree on the plot is hierarchical clustering of NCBITaxon based on shared ASVs. Categories with no branch length between them are subsets of each other. The fraction next to the category name indicates the fraction of 'good' ASVs in it; 'good' ASVs are the ones exceeding the arbitrary absValue cutoff (option in taxon_mwuPlot()). For Fisher's based test, specify absValue=0.5. This value does not affect statistics and is used for plotting only. The original idea was for genes differential expression analysis from Wright et al (2015)

<doi:10.1186/s12864-015-1540-2>; adapted here for taxonomic analysis. The 'Anaconda' package makes it possible to carry out these analyses by automatically creating several graphs and tables and storing them in specially created subfolders. You will need your QIIME2 pipeline output for each kingdom (eg; Fungi and/or Bacteria and/or Metazoan): i) taxonomy.tsv, ii) taxonomy_RepSeq.tsv, iii) ASV.tsv and iv) SampleSheet_comparison.txt (the latter being created by you).

License GPL (>= 2)

URL `https://github.com/PLStenger/Anaconda`

BugReports `https://github.com/PLStenger/Anaconda/issues`

Imports ggrepel, pheatmap, lookup, plyr, data.table, rafalib, RColorBrewer, methods, graphics

Depends ape, DESeq2, ggplot2

Encoding UTF-8

RoxygenNote 7.2.1

NeedsCompilation no

Config/pak/sysreqs libssl-dev

Repository https://plstenger.r-universe.dev

RemoteUrl https://github.com/plstenger/anaconda

RemoteRef HEAD

RemoteSha 75f263c08aaa003cc03743c8ed839c2df653bc41

Contents

`Bacteria` *Bacteria*

Description

This function create a new folder named Bacteria and set your working directory into this folder. Please, run setwd("Bacteria") after this function.

Usage

```
Bacteria(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, a new folder named Bacteria will be created and your working directory will be set into this folder, depending of the selected Kingdom.

Value

A new folder named Bacteria will be created and your working directory will be set into this folder, depending of the selected Kingdom.

Examples

```
## Not run: Bacteria()
# Please, run setwd("Bacteria") after this function.
```

`clusteringGOs` *clusteringGOs*

Description

clusteringGOs from DESeq2 analysis pipeline

Usage

```
clusteringGOs(gen2go, div, cutHeight)
```

Arguments

`gen2go`	from DESeq2 analysis pipeline
`div`	div
`cutHeight`	cutHeight

Value

a clustering GO

Examples

```
## Not run: clusteringGOs()
```

`dasva_raw_input` *dasva_raw_input*

Description

Used in heatmap_samples_hclust(), heatmap_samples_matrix(), PCA_data_dasva() and get_dasva() functions.

Usage

```
dasva_raw_input(sampleTable, directory = ".", design, ignoreRank = FALSE, ...)
```

Arguments

`sampleTable`	Depending of the heatmap_samples_hclust(), heatmap_samples_matrix(), PCA_data_dasva() and get_dasva() functions.
`directory`	directory
`design`	design
`ignoreRank`	ignoreRank
`...`	...

Value

object

Examples

```
  ## Not run: dasva_raw <- dasva_raw_input(sampleTable = sampleTable,
  directory = targeted_analysis_dir,
  design= ~ condition)
## End(Not run)
```

`database_bacteria_creation` *database_bacteria_creation*

Description

Create a Database for Bacteria kingdom for Global analysis by Taxon_MWU analysis from targeted analysis. Please, run setwd("02_Global_analysis") after this function.

Usage

```
database_bacteria_creation(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, the database will create itself.

Value

A data frame file named database_bacteria_package_all.tab created from the taxonomy_all_bacteria_QIIME2_and_NCBI_fo file and your own taxonomy_RepSeq.tsv file. database_bacteria_creation()

Examples

```
# It is important not to write anything between the brackets, the database will create itself.
## Not run: database_bacteria_creation()
# Please, run setwd("02_Global_analysis") after this function.
```

`database_fungi_creation`

database_fungi_creation

Description

Create a Database for Fungi kingdom for Global analysis by Taxon_MWU analysis from targeted analysis only from rarefied ASVs. Please, run setwd("02_Global_analysis") after this function.

Usage

```
database_fungi_creation(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, the database will create itself.

Value

A data frame file named database_fungi_package_all.tab created from the taxonomy_all_bacteria_QIIME2_and_NCBI_form file and your own taxonomy.tsv file.

Examples

```
# It is important not to write anything between the brackets, the database will create itself.
## Not run: database_fungi_creation()
# Please, run setwd("02_Global_analysis") after this function.
```

`database_fungi_creation_RepSeq`

database_fungi_creation_RepSeq

Description

Create a Database for Fungi kingdom for Global analysis by Taxon_MWU analysis from targeted analysis. Please, run setwd("02_Global_analysis") after this function.

Usage

```
database_fungi_creation_RepSeq(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, the database will create itself.

Value

A data frame file named database_fungi_package_all.tab created from the taxonomy_all_bacteria_QIIME2_and_NCBI_form
file and your own taxonomy_RepSeq.tsv file.

Examples

```
# It is important not to write anything between the brackets, the database will create itself.
## Not run: database_fungi_creation_RepSeq()
# Please, run setwd("02_Global_analysis") after this function.
```

`database_metazoan_creation`

database_metazoan_creation

Description

Create a Database for metazoan kingdom for Global analysis by Taxon_MWU analysis from targeted analysis. Please, run setwd("02_Global_analysis") after this function.

Usage

```
database_metazoan_creation(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, the database will create itself.

Value

A data frame file named database_metazoan_package_all.tab created from the taxonomy_all_metazoan_QIIME2_and_NCBI
file and your own taxonomy_RepSeq.tsv file. database_metazoan_creation()

Examples

```
# It is important not to write anything between the brackets, the database will create itself.
## Not run: database_metazoan_creation()
# Please, run setwd("02_Global_analysis") after this function.
```

`fisherTest` *fisherTest*

Description

Ficher Test from RBGOA

Usage

```
fisherTest(gotable)
```

Arguments

`gotable`	from gomwuStats from RBGOA

Value

fisherTest

Examples

```
## Not run: fisherTest()
```

`format_input` *format_input*

Description

Apply logP on both positive and negative ASVs FC

Usage

```
format_input(x)
```

Arguments

`x`	Object from the Differential ASV abundance (DASVA) analysis

Value

an input for the input_global_analysis() function

Examples

```
## Not run: format_input(x)
```

`Fungi` *Fungi*

Description

This function create a new folder named Fungi and set your working directory into this folder. Please, run setwd("Fungi") after this function.

Usage

```
Fungi(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, a new folder named Fungi will be created and your working directory will be set into this folder, depending of the selected Kingdom.

Value

A new folder named Fungi will be created and your working directory will be set into this folder, depending of the selected Kingdom.

Examples

```
## Not run: Fungi()
# plaese, run setwd("Fungi") after this function.
```

`funguild_input_targeted`
funguild_input_targeted

Description

Prepare Object for Fungi Guilds for Fungi kingdom for targeted analysis

Usage

```
funguild_input_targeted(x)
```

Arguments

`x`	Object from the Differential ASV abundance (DASVA) analysis

Value

An Object used for Fungi Guilds informations for Fungi kingdom for targeted analysis from the Differential ASV abundance (DASVA) analysis

Examples

```
## Not run: get_funguilds_targeted(res_forest_vs_long_fallow_guilds)
```

`get_bactotraits_targeted` *get_bactotraits_targeted*

Description

Obtain Bacterial Traits for Bacteria kingdom for targeted analysis

Usage

```
get_bactotraits_targeted(x)
```

Arguments

`x`	Object from the Differential ASV abundance (DASVA) analysis

Value

A data frame file with Bacterial Traits informations for Bacteria kingdom for targeted analysis from the Differential ASV abundance (DASVA) analysis

Examples

```
## Not run: get_bactotraits_targeted(res_forest_vs_long_fallow)
```

`get_dasva` *get_dasva*

Description

Creates the DASVA object. Fit a Gamma-Poisson Generalized Linear Model, dispersion estimates for Negative Binomial distributed data, "parametric", local" or "mean"

Usage

```
get_dasva(fitType = "")
```

Arguments

`fitType`	Fit a Gamma-Poisson Generalized Linear Model, dispersion estimates for Negative Binomial distributed data, "parametric", local" or "mean"

Value

DASVA object

Examples

```
## Not run: dasva <- get_dasva(fitType="parametric")
dasva <- get_dasva(fitType="local")
dasva <- get_dasva(fitType="mean")
## End(Not run)
```

`get_funguilds` *get_funguilds*

Description

get Fungi Guilds from taxon_list_drawer Object

Usage

```
get_funguilds(taxon_list_drawer)
```

Arguments

`taxon_list_drawer`	object from get_taxon_list_drawer() function

Value

funguilds Object

Examples

```
## Not run: funguilds <- get_funguilds(taxon_list_drawer)
```

`get_funguilds_targeted` *get_funguilds_targeted*

Description

Obtain Fungi Guilds for Fungi kingdom for targeted analysis

Usage

```
get_funguilds_targeted(x)
```

Arguments

`x`	Object from the funguild_input_targeted() output.

Value

A data frame file with Fungi Guilds informations for Fungi kingdom for targeted analysis from the Differential ASV abundance (DASVA) analysis

Examples

```
## Not run: get_funguilds_targeted(res_forest_vs_long_fallow_guilds)
```

`get_input_files` *get_input_files*

Description

Created sub directory "Targeted_analysis" if not already exist. Then, create one file by condition into it, and then upload the taxonomy file. Please, run setwd("01_Targeted_analysis") after this function.

Usage

```
get_input_files(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, all inputs will be adapted automatically.

Value

taxo

Examples

```
## Not run: taxo <- get_input_files()
# please, run setwd("01_Targeted_analysis") after this function.
```

`get_link_guilds` *get_link_guilds*

Description

get link guilds from taxon_list and funguilds Objects

Usage

```
get_link_guilds(taxon_list, funguilds)
```

Arguments

`taxon_list`	object from taxon_mwu_list() function
`funguilds`	object from get_funguilds() function

Value

link_guilds Object

Examples

```
## Not run: link_guilds <- get_link_guilds(taxon_list, funguilds)
```

`get_taxon_list_drawer` *get_taxon_list_drawer*

Description

get taxonomic list drawer

Usage

```
get_taxon_list_drawer(taxon_list)
```

Arguments

`taxon_list`	object from taxon_mwu_list() function

Value

taxon_list_drawer Object and "taxon_list_drawer_input.txt" file

Examples

```
## Not run: taxon_list_drawer <- get_taxon_list_drawer(taxon_list)
```

`heatmap_condition` *heatmap_condition*

Description

For Clustering step. Fill directly the annotation_col variable of the pheatmap() function

Usage

```
heatmap_condition(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, all inputs will be adapted automatically.

Value

Fill directly the annotation_col variable of the pheatmap() function

---

`heatmap_data_dasva` *heatmap_data_dasva*

Description

For Clustering step. Create the log2.norm.counts object.

Usage

```
heatmap_data_dasva(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, all inputs will be adapted automatically.

Value

Create the log2.norm.counts object.

`heatmap_samples_hclust`

heatmap_samples_hclust

Description

Adapt hclust for heatmap sample to sample analysis

Usage

```
heatmap_samples_hclust(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, all inputs will be adapted automatically.

Value

hclust object for the heatmap.2() function

Examples

```
## Not run: hc <- heatmap_samples_hclust()
```

`heatmap_samples_matrix`

heatmap_samples_matrix

Description

Adapt samples matrix for heatmap sample to sample analysis

Usage

```
heatmap_samples_matrix(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, all inputs will be adapted automatically.

Value

samples matrix object for the heatmap.2() function

Examples

```
## Not run: mat <- heatmap_samples_matrix()
```

`heatmap_taxo` *heatmap_taxo*

Description

Adding taxonomy in the pheatmap plot, instead of ASVs codes

Usage

```
heatmap_taxo(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, all inputs will be adapted automatically.

Value

log2.norm.counts_taxo used fro adding taxonomy in the pheatmap plot, instead of ASVs codes

Examples

```
## Not run: log2.norm.counts_taxo <- heatmap_taxo()
```

`input_global_analysis` *input_global_analysis*

Description

Input files creation for each condition for Global analysis by Taxon_MWU analysis from targeted analysis (I)

Usage

```
input_global_analysis(x)
```

Arguments

`x`	Object from the Differential ASV abundance (DASVA) analysis

Value

Input Object for Global analysis by Taxon_MWU analysis from targeted analysis (I)

Examples

```
## Not run: input_global_analysis(res_forest_vs_long_fallow)
```

`Metazoan` *Metazoan*

Description

This function create a new folder named Metazoan and set your working directory into this folder. Please, run setwd("Metazoan") after this function.

Usage

```
Metazoan(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, a new folder named Metazoan will be created and your working directory will be set into this folder, depending of the selected Kingdom.

Value

A new folder named Metazoan will be created and your working directory will be set into this folder, depending of the selected Kingdom.

Examples

```
## Not run: Metazoan()
# plaese, run setwd("Metazoan") after this function.
```

`move_files` *move_files*

Description

Move the file in the good folders. Depending on the previous Kingdom selection (e.g., Fungi 'Fungi()', Bacteria 'Bacteria()', etc.)

Usage

```
move_files(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, files will move in the good folders, depending of the selected Kingdom before.

Value

Move the file in the good folders.

Examples

```
## Not run: move_files()
```

`mwuTest` *mwuTest*

Description

Mann-Whitney U Test from RBGOA

Usage

```
mwuTest(gotable, Alternative)
```

Arguments

`gotable`	from gomwuStats from RBGOA
`Alternative`	from gomwuStats from RBGOA

Value

mwuTest

Examples

```
## Not run: mwuTest()
```

`PCA_data_dasva` *PCA_data_dasva*

Description

Compute the PCA (Pincipal Component Analysis) data.

Usage

```
PCA_data_dasva(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, all inputs will be adapted automatically.

Value

data. The PCA (Pincipal Component Analysis) data.

`plotDispASVs` *plotDispASVs*

Description

Create a plot of Dispersion ASV

Usage

```
plotDispASVs(
  object,
  ymin,
  CV = FALSE,
  genecol = "black",
  fitcol = "red",
  finalcol = "dodgerblue",
  legend = TRUE,
  xlab,
  ylab,
  log = "xy",
  cex = 0.45,
  ...
)
```

Arguments

`object`	Corresponding to the DASVA (Differential ASV abundance) object
`ymin`	ymin
`CV`	CV
`genecol`	genecol
`fitcol`	fitcol
`finalcol`	finalcol
`legend`	legend
`xlab`	xlab
`ylab`	ylab
`log`	log
`cex`	cex
`...`	...

Value

A plot of Dispersion ASV

Examples

```
## Not run: plotDispASVs(dasva)
```

`plotMA.dasva` *plotMA.dasva*

Description

Custom MA plots for the Differential ASV abundance (DASVA) analysis. defining a new function to plot all ASVs and not only log2FoldChange > 2

Usage

```
plotMA.dasva(
  object,
  alpha,
  main = "",
  xlab = "mean of normalized counts",
  ylim,
  MLE = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

`object`	Object from the Differential ASV abundance (DASVA) analysis
`alpha`	alpha
`main`	main
`xlab`	xlab
`ylim`	ylim
`MLE`	MLE
`verbose`	verbose
`...`	...

Value

A MA plot

Examples

```
## Not run: plotMA.dasva(rXXX, main="XXX", ylim=c(-20,20))
```

`plotPCA.san` *plotPCA.san*

Description

Custom plotPCA function to plot PC1 et PC3

Usage

```
plotPCA.san(object, intgroup = "condition", ntop = 500, returnData = FALSE)
```

Arguments

`object`	An object use for the PCA
`intgroup`	intgroup
`ntop`	ntop
`returnData`	returnData

Value

A PCA

Examples

```
## Not run: plotPCA.san(object)
```

`plotSparsityASV` *plotSparsityASV*

Description

Create a plot of Sparsity ASV

Usage

```
plotSparsityASV(x, normalized = TRUE, ...)
```

Arguments

`x`	Corresponding to the DASVA (Differential ASV abundance) object
`normalized`	normalized
`...`	...

Value

A plot of Sparsity ASV

Examples

```
## Not run: plotSparsityASV(dasva)
```

`samplesInfo` *samplesInfo*

Description

Imports conditions information from your SampleSheet_comparison.txt file, with focus on samplesInfo.

Usage

```
samplesInfo(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, comparisons will create themselves.

Value

a data.frame with conditions information from your SampleSheet_comparison.txt file, with focus on samplesInfo.

Examples

```
## Not run: samplesInfo <- samplesInfo()
```

`target_file` *target_file*

Description

Imports conditions information from your SampleSheet_comparison.txt file, with focus on iput files.

Usage

```
target_file(nothing)
```

Arguments

`nothing`	It's important not to write anything between the brackets, comparisons will create themselves.

Value

a data.frame with conditions information from your SampleSheet_comparison.txt file

Examples

```
## Not run: target_file <- target_file()
```

`taxon_mwuPlot` *taxon_mwuPlot*

Description

taxon mwuPlot for taxonomic analysis

Usage

```
taxon_mwuPlot(
  inFile,
  goAnnotations,
  goDivision,
  level1 = 0.1,
  level2 = 0.05,
  level3 = 0.01,
  absValue = -log(0.05, 10),
  adjusted = TRUE,
  txtsize = 1,
  font.family = "sans",
  treeHeight = 0.5,
  colors = NULL,
  verbose = TRUE
)
```

Arguments

`inFile`	inFile - results object from the DASVA analysis
`goAnnotations`	parallel to goAnnotations from gomwuStats from RBGOA. Here, "database_bacteria_package_all.tab" if Bacteria, "database_fungi_package_all.tab" if Fungi
`goDivision`	parallel to goAnnotations from gomwuStats from RBGOA. Here, "TR" = taxonomic Rank, don't change this
`level1`	level1
`level2`	level2

`level3`	level3
`absValue`	absValue
`adjusted`	adjusted
`txtsize`	txtsize
`font.family`	font.family
`treeHeight`	treeHeight
`colors`	colors
`verbose`	verbose

Value

taxon mwuPlot and goods "Table_02_taxon_mwuPlot.txt"

Examples

```
## Not run: taxon_mwuPlot(input,...)
```

`taxon_mwuPlot_guilds` *taxon_mwuPlot_guilds*

Description

taxon Mann-Whitney U Plot with Fungi Guilds added

Usage

```
taxon_mwuPlot_guilds(
  inFile,
  goAnnotations,
  goDivision,
  level1 = 0.1,
  level2 = 0.05,
  level3 = 0.01,
  absValue = -log(0.05, 10),
  adjusted = TRUE,
  txtsize = 1,
  font.family = "sans",
  treeHeight = 0.5,
  colors = NULL,
  verbose = TRUE
)
```

Arguments

`inFile`	inFile - results object from the DASVA analysis
`goAnnotations`	parallel to goAnnotations from gomwuStats from RBGOA. Here, "database_bacteria_package_all.tab" if Bacteria, "database_fungi_package_all.tab" if Fungi
`goDivision`	parallel to goAnnotations from gomwuStats from RBGOA. Here, "TR" = taxonomic Rank, don't change this
`level1`	level1
`level2`	level2
`level3`	level3
`absValue`	absValue
`adjusted`	adjusted
`txtsize`	txtsize
`font.family`	font.family
`treeHeight`	treeHeight
`colors`	colors
`verbose`	verbose

Value

List for the statistical analysis for taxonomic rank

Examples

```
## Not run: taxon_mwuPlot_guilds(input, ...)
```

`taxon_mwuStats` *taxon_mwuStats*

Description

mwuStats from RBGOA adapted for taxonomic analysis

Usage

```
taxon_mwuStats(
  input,
  goDatabase,
  goAnnotations,
  goDivision,
  Alternative = "t",
  adjust.multcomp = "BH",
  clusterCutHeight = 0.25,
  largest = 0.1,
```

```
  smallest = 5,
  perlPath = "perl",
  verbose = TRUE
)
```

Arguments

`input`	input
`goDatabase`	goDatabase
`goAnnotations`	goAnnotations
`goDivision`	goDivision
`Alternative`	Alternative
`adjust.multcomp`	adjust.multcomp
`clusterCutHeight`	clusterCutHeight
`largest`	largest
`smallest`	smallest
`perlPath`	perlPath
`verbose`	verbose

Value

Statistical analysis for taxonomic rank

Examples

```
## Not run: taxon_mwuStats(input, ...)
```

`taxon_mwu_list` *taxon_mwu_list*

Description

taxon Mann-Whitney U list for taxonomic analysis

Usage

```
taxon_mwu_list(
  inFile,
  goAnnotations,
  goDivision,
  level1 = 0.1,
  level2 = 0.05,
  level3 = 0.01,
```

```
  absValue = -log(0.05, 10),
  adjusted = TRUE,
  txtsize = 1,
  font.family = "sans",
  treeHeight = 0.5,
  colors = NULL
)
```

Arguments

`inFile`	inFile - results object from the DASVA analysis
`goAnnotations`	parallel to goAnnotations from gomwuStats from RBGOA. Here, "database_bacteria_package_all.tab" if Bacteria, "database_fungi_package_all.tab" if Fungi
`goDivision`	parallel to goAnnotations from gomwuStats from RBGOA. Here, "TR" = taxonomic Rank, don't change this
`level1`	level1
`level2`	level2
`level3`	level3
`absValue`	absValue
`adjusted`	adjusted
`txtsize`	txtsize
`font.family`	font.family
`treeHeight`	treeHeight
`colors`	colors

Value

List for the statistical analysis for taxonomic rank

Examples

```
## Not run: taxon_list <- taxon_mwu_list(input, ...)
```

Index