Package 'macrosyntR'

March 29, 2023

Type Package

Title Draw Ordered Oxford Grids

Version 0.2.19

Maintainer Sami El Hilali <elhilali.sami@gmail.com>

Depends R (>= 4.1.0)

Imports stats, utils, ggplot2, ggthemes, igraph, tidyr, reshape2, dplyr, stringr

Description Use standard genomics file format (BED) and a table of orthologs to illustrate pair-wise synteny conservation at the genome-wide scale. Significantly conserved linkage groups are identified as described in Simakov et al. (2020) <doi:10.1038/s41559-020-1156-z> and displayed on an Oxford Grid (Edwards (1991) <doi:10.1111/j.1469-1809.1991.tb00394.x>). The package provides a function that uses a network-based greedy algorithm to find communities (Clauset et al. (2004) <doi:10.1103/PhysRevE.70.066111>) and so automatically order the chromosomes on the plot to improve interpretability.

Encoding UTF-8

License GPL-3

URL https://github.com/SamiLhll/macrosyntR

BugReports https://github.com/SamiLhll/macrosyntR/issues

RoxygenNote 7.2.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Sami El Hilali [aut, cre] (<https://orcid.org/0000-0003-4417-8399>), Richard Copley [aut] (<https://orcid.org/0000-0001-7846-4954>)

Repository CRAN

Date/Publication 2023-03-29 13:30:16 UTC

R topics documented:

`compute_macrosynteny` *Compute significant macrosynteny blocks*

Description

This is a function to generate the contingency table of an orthologs dataframe and apply fischer test to calculate the significant associations. It outputs a dataframe shaped as following : sp1.Chr,sp2.Chr,a,pval,significant,pval_a

Usage

```
compute_macrosynteny(orthologs_df, pvalue_threshold = 0.001)
```

Arguments

`orthologs_df`	dataframe. orthologs with genomic coordinates loaded with load_orthologs()
`pvalue_threshold`	numeric. threshold for significancy. (default equals 0.001)

Value

A dataframe object

Examples

```
# basic usage of compute_macrosynteny :

orthologs_table <- system.file("extdata","my_orthologs.tab",package="macrosyntR")

my_orthologs <- read.table(orthologs_table,header=TRUE)

my_macrosynteny <- compute_macrosynteny(my_orthologs)
```

get_syntenic_genes *get the syntenic genes as a table*

Description

This is a function to extract all the syntenic genes from an orthologs_df. It requires as input an orthologs_df loaded by load_orthologs().

Usage

```
get_syntenic_genes(orthologs_df)
```

Arguments

orthologs_df	dataframe. orthologs with genomic coordinates loaded by load_orthologs()

Value

dataframe composed of details for each detected syntenic block of genes. It contains the following columns : sp1.Chr, sp1.Start, sp1.End, sp2.Chr, sp2.Start, sp2.End, size, sp1.IDs, sp2.IDs

See Also

load_orthologs()

Examples

```
# basic usage of get_syntenic_genes :

orthologs_file <- system.file("extdata","Bflo_vs_Pech.tab",package="macrosyntR")
bedfile_sp1 <- system.file("extdata","Bflo.protein_products.bed",package="macrosyntR")
bedfile_sp2 <- system.file("extdata","Pech.protein_products.bed",package="macrosyntR")

my_orthologs <- load_orthologs(orthologs_table = orthologs_file,
                               sp1_bed = bedfile_sp1,
                               sp2_bed = bedfile_sp2)

my_syntenic_block_of_genes <- get_syntenic_genes(my_orthologs)
```

load_orthologs *load orthologs with their genomic coordinates.*

Description

Puts together the table of orthologous genes with their genomic coordinates in the two species under study. It outputs a data.frame shaped as following : sp1.ID,sp1.Chr,sp1.Start,sp1.End,sp1.Index,sp2.ID,sp2.Chr,sp2.Start,sp2.

Usage

```
load_orthologs(orthologs_table, sp1_bed, sp2_bed)
```

Arguments

orthologs_table	character. Full path to the orthologs table (format : geneID_on_species1 geneID_on_species2)
sp1_bed	character. Full path to the genomic coordinates of the genes on species1 (BED format)
sp2_bed	character. Full path to the genomic coordinates of the genes on species2 (BED format)

Value

dataframe composed of genomic coordinates and relative index of orthologs on both species

Examples

```
# basic usage of load_orthologs :

orthologs_file <- system.file("extdata","Bflo_vs_Pech.tab",package="macrosyntR")
bedfile_sp1 <- system.file("extdata","Bflo.protein_products.bed",package="macrosyntR")
bedfile_sp2 <- system.file("extdata","Pech.protein_products.bed",package="macrosyntR")

my_orthologs <- load_orthologs(orthologs_table = orthologs_file,
                               sp1_bed = bedfile_sp1,
                               sp2_bed = bedfile_sp2)
```

plot_macrosynteny *Plot Macro-synteny*

Description

This is a function to generate the contingency table of an MBH dataframe and apply fischer test to calculate the significant associations.

Usage

```
plot_macrosynteny(macrosynt_df, sp1_label = "", sp2_label = "")
```

Arguments

macrosynt_df	dataframe of contingency table with p-values calculated by the compute_macrosynteny() function
sp1_label	character. The name of the species1 to display on the plot
sp2_label	character. The name of the species2 to put on the plot

Value

ggplot2 object

See Also

compute_macrosynteny()

Examples

```
# basic usage of plot_macrosynteny :

orthologs_table <- system.file("extdata","my_orthologs.tab",package="macrosyntR")

my_orthologs <- read.table(orthologs_table,header=TRUE)

my_macrosynteny <- compute_macrosynteny(my_orthologs)

plot_macrosynteny(my_macrosynteny,
                  sp1_label = "B.floridae",
                  sp2_label = "P.echinospica")
```

---

plot_oxford_grid *plot the Macro-synteny as an oxford grid.*

Description

This is a function to plot the oxford grided plot to compare the macro synteny of two species. It requires as input an orthologs_df loaded by load_orthologs()

Usage

```
plot_oxford_grid(
  orthologs_df,
  sp1_label = "",
  sp2_label = "",
  dot_size = 0.5,
  dot_alpha = 0.4,
  reorder = FALSE,
  keep_only_significant = FALSE,
  color_by = NULL,
  pvalue_threshold = 0.001,
  color_palette = NULL,
  shade_non_significant = TRUE,
  reverse_species = FALSE,
  keep_sp1_raw_order = FALSE
)
```

Arguments

orthologs_df	dataframe. orthologs with genomic coordinates loaded by the load_orthologs()
sp1_label	character. name of 1st species to display on the plot
sp2_label	character. name of 2nd species to display on the plot
dot_size	numeric. (default = 0.5)
dot_alpha	numeric. (default = 0.4)
reorder	logical. (default = FALSE) tells whether to reorder the chromosomes in clusters as implemented in reorder_macrosynteny()
keep_only_significant	logical. (default = FALSE)
color_by	string/variable name. (default = NULL) column of the orthologs_df to use to color the dots.
pvalue_threshold	numeric. (default = 0.001)
color_palette	vector. (default = NULL) list of colors (as string under double quote) for the clusters. The amount of colors must match the amount of clusters.
shade_non_significant	logical. (default = TRUE) When TRUE the orthologs located on non-significant linkage groups are displayed in "grey"
reverse_species	logical. (default = FALSE) When TRUE the x and y axis of the plot are reversed. sp1 is displayed on the y axis and sp2 is displayed on the x axis.
keep_sp1_raw_order	logical.(default equals FALSE) tells if the reordering should be constrained on the species1 and change just the order of the species2

Value

A ggplot2 object

See Also

load_orthologs()

reorder_macrosynteny()

Examples

```
# basic usage of plot_oxford_grid :

orthologs_table <- system.file("extdata","my_orthologs.tab",package="macrosyntR")

my_orthologs <- read.table(orthologs_table,header=TRUE)

plot_oxford_grid(my_orthologs,
                 sp1_label = "B.floridae",
                 sp2_label = "P.echinospica")

# plot a reordered Oxford Grid and color by cluster :

plot_oxford_grid(my_orthologs,
                 sp1_label = "B.floridae",
                 sp2_label = "P.echinospica",
                 reorder = TRUE,
                 color_by = "clust")
```

---

reorder_macrosynteny *Reorder the mbh_df before plotting*

Description

This is a function to reorder an orthologs_df, that was generated with load_orthologs(). It retrieves communities using igraph::cluster_fast_greedy.

Usage

```
reorder_macrosynteny(
  orthologs_df,
  pvalue_threshold = 0.001,
  keep_only_significant = FALSE,
  keep_sp1_raw_order = FALSE
)
```

Arguments

orthologs_df	dataframe. mutual best hits with genomic coordinates loaded with load_orthologs()
pvalue_threshold	numeric. threshold for significancy. (default equals 0.001)

keep_only_significant
: logical. (default equals FALSE) tells if the non significant linkage groups should be removed. It drastically speeds up the computation when using one highly fragmented genome.

keep_sp1_raw_order
: logical. (default equals FALSE) tells if the reordering should be constrained on the species1 and change just the order of the species2

Value

A dataframe object

See Also

load_orthologs()

compute_macrosynteny()

Examples

```
# basic usage of reorder_macrosynteny :

orthologs_table <- system.file("extdata","my_orthologs.tab",package="macrosyntR")

my_orthologs <- read.table(orthologs_table,header=TRUE)

my_orthologs_reordered <- reorder_macrosynteny(my_orthologs)
```

---

reverse_species_order *Reverse order of the species in an orthologs_df.*

Description

Returns an orthologs_df (data.frame) with reversed species order compared to the inputted orthologs_df. sp1 becomes sp2 and the otherway around. It intends at facilitating the integration of more than just two datasets. It outputs a data.frame shaped as following : sp1.ID,sp1.Chr,sp1.Start,sp1.End,sp1.Index,sp2.ID,s

Usage

```
reverse_species_order(orthologs_df)
```

Arguments

orthologs_df
: orthologs_df dataframe. mutual best hits with genomic coordinates loaded with load_orthologs()

Value

dataframe composed of genomic coordinates and relative index of orthologs on both species

See Also

load_orthologs()

Examples

```
# basic usage of reverse_species_order :

orthologs_file <- system.file("extdata","Bflo_vs_Pech.tab",package="macrosyntR")
bedfile_sp1 <- system.file("extdata","Bflo.protein_products.bed",package="macrosyntR")
bedfile_sp2 <- system.file("extdata","Pech.protein_products.bed",package="macrosyntR")

my_orthologs <- load_orthologs(orthologs_table = orthologs_file,
                               sp1_bed = bedfile_sp1,
                               sp2_bed = bedfile_sp2)
my_orthologs_reversed <- reverse_species_order(my_orthologs)
```

Index