Package 'ggrasp'

July 1, 2018

Type Package

Title Gaussian-Based Genome Representative Selector with Prioritization

Version 1.0

Description Given a group of genomes and their relationship with each other, the package clusters the genomes and selects the most representative members of each cluster. Additional data can be provided to the prioritize certain genomes. The results can be printed out as a list or a new phylogeny with graphs of the trees and distance distributions also available. For detailed introduction see: Thomas H Clarke, Lauren M Brinkac, Granger Sutton, and Derrick E Fouts (2018), GGRaSP: a R-package for selecting representative genomes using Gaussian mixture models, Bioinformatics, bty300, <doi:10.1093/bioinformatics/bty300>.

Depends R(>= 3.1.0)

Imports ggplot2, mixtools, ape, bgmm, colorspace, methods

License GPL-2

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Thomas Clarke [aut, cre]

Maintainer Thomas Clarke <tclarke@jcvi.org>

Repository CRAN

Date/Publication 2018-07-01 13:30:35 UTC

R topics documented:

`ggrasp-class` *An S4 class representing the GGRaSP data and output*

Description

An S4 class representing the GGRaSP data and output

Slots

`dist.mst` The distance matrix showing the distances between different genomes

`phy` The phylogenetic tree in newick format

`rank` The ranks of the respective genomes with lower getting higher priority in being called as a medoid

`cluster` A vector giving the numeric cluster ID for each genome

`h` The threshold variable used to make the clusters

`medoids` A vector giving the medoid for each cluster

`gmm` A data.frame containing all the gaussian distributions used to find the threshold when available

`gmm.orig` A data.frame containing all the gaussian distributions prior to cleaning. Used to recalculate the threshold when needed

---

`ggrasp.addRanks` *ggrasp.addRanks*

Description

adds a rank file to a GGRaSP object. If clusters have been defined, the medoids will be re-defined

Usage

```
ggrasp.addRanks(x, rank.file)
```

Arguments

`x`	the GGRaSP object for which the ranks will be added.
`rank.file`	string pointing to a file containing the ranks

Value

A GGRaSP object where the ranks have been entirely redefined with the ranks in rank.file

ggrasp.cluster *ggrasp.cluster*

Description

ggrasp.cluster() clusters the genomes in a GGRaSP class variable and assigns the most representative genome in each cluster after accounting for rank as a medoid.

Usage

```
ggrasp.cluster(ggrasp.data, threshold, num.clusters, z.limit = 1,
  gmm.start = 2, gmm.max = 10, min.lambda = 0.005, run.type = "bgmm",
  left.dist = 1)
```

Arguments

ggrasp.data	Required. If neither a threshold or a num.cluster is given, a mixed model of Gaussian distributions is used to estimate a threshold to use the cluster.
threshold	The threshold used to cluster together all genomes within this distance.
num.clusters	Create this number of clusters independent of the cluster.
z.limit	All Gaussian distributions with means within this number of standard deviations will be reduced to only the larger distribution. Defaults to 1. Set to 0 to keep all non-overlapping distributions.
gmm.start	Number of Gaussian distributions to start the examination. Must be at least 2 and not greater than the gmm.max.
gmm.max	Maximum number of Gaussian distributions to examine. Has to be at least 2. 10 is the default
min.lambda	All Gaussian distributions with lambda value (proportion of the total distribution) below this value are removed before calculating the threshold. Default is 0.005. Set to 0 to keep all.
run.type	String giving the package to use to get the mixture model. Currently "bgmm" (default) and mixtools" are implemented.
left.dist	Number giving the number Gaussian distribution model immediately to the left of the threshold used. 1 is the default. Only value between 1 and k-1 where k is the total number of number of Gaussian distributions.

Value

Returns a class GGRaSP variable with the clusters and medoids assigned. In cases where the Gaussian Mixture Model was used to estimate the cutoff threshold, the descriptive values of the different distributions is also stored

ggrasp.load *ggrasp.load*

Description

ggrasp.load() initializes a class GGRaSP object from a file containing either a tree, a distance matrix or a multi-fasta alignment. The returned object can subsequently be clustered using ggrasp.cluster().

Usage

```
ggrasp.load(file, file.format, rank.file, offset, tree.method = "complete")
```

Arguments

file	File containing the tree, matrix or sequence alignment used to initialize the ggrasp object. Required.
file.format	The format the file is in, with tree, fasta and matrix accepted. If not given the program makes a guess.
rank.file	File containing the ranks of genomes in a tab-delineated file with the genome in column 1 and the rank in column 2. The rank is a non-negative number.
offset	Numeric representing a perfect match. Default is 0.
tree.method	The method used to make the tree from a distance matrix. "Complete" (Default), "Average", "Single", and "nj" (Neighbor Joining) are currently available.

Value

Returns a class GGRaSP variable

Examples

```
#The following data is from Chavda et al 2016 which phylotyped Enterobacter genomes
# Our example uses the data underpinning the tree shown in Figure 2
# Also included is a ranking file to prioritize closed Enterobactor genomes

library(ggrasp);
tree.file <- system.file("extdata", "Enter.kSNP.tree", package="ggrasp")
rank.file.in <- system.file("extdata", "Enter.kSNP.ranks", package="ggrasp")
Enter.tree <- ggrasp.load(tree.file, file.format = "tree", rank.file = rank.file.in);

# Other options include loading by fasta file:
fasta.file <- system.file("extdata", "Enter.kSNP2.fasta", package="ggrasp")
rank.file.in <- system.file("extdata", "Enter.kSNP.ranks", package="ggrasp")
Enter.tree <- ggrasp.load(fasta.file, file.format = "fasta", rank.file =rank.file.in)

# and by distance matrix. Since this distance matrix is actually percent identity,
# we will us an offset of 100
mat.file <- system.file("extdata", "Enter.ANI.mat", package="ggrasp")
rank.file.in <- system.file("extdata", "Enter.kSNP.ranks", package="ggrasp")
```

```
Enter.in <- ggrasp.load(mat.file, file.format = "matrix", rank.file =rank.file.in, offset = 100)

# Use summary() to examine the data loaded
summary(Enter.in)

#Use plot() to see the tree
plot(Enter.in)
```

---

ggrasp.recluster *ggrasp.recluster*

Description

recalculates a threshold and the resulting cluster using the previously defined Gaussian Mixture Model and provided threshold-determining factors. Requires the ggrasp.cluster to already have run

Usage

```
ggrasp.recluster(x, z.limit = 1, min.lambda = 0.005, left.dist = 1)
```

Arguments

x	the GGRaSP object for which the ranks will be added.
z.limit	All Gaussian distributions with means within this number of standard deviations will be reduced to only the larger distribution. Defaults to 1. Set to 0 to keep all non-overlapping distributions.
min.lambda	All Gaussian distributions with lambda value (proportion of the total distribution) below this value are removed before calculating the threshold. Default is 0.005. Set to 0 to keep all.
left.dist	Number giving the number Gaussian distribution model immediately to the left of the threshold used. 1 is the default. Only value between 1 and k-1 where k is the total number of number of Gaussian distributions.

Value

A GGRaSP object with the recalculated thresholds and the medoids using a previously generated GMM

Examples

```
#The following data is from Chavda et al 2016 which phylotyped Enterobacter genomes
# Our example uses the data underpinning the tree shown in Figure 2

#Loading the tree
library(ggrasp);
tree.file <- system.file("extdata", "Enter.kSNP.tree", package="ggrasp")
Enter.tree <- ggrasp.load(tree.file, file.format = "tree");
```

```
#Clustering the tree using a threshold estimated by Gaussian Mixture Models (GMMs)
Enter.tree.cluster <- ggrasp.cluster(Enter.tree)


#Use print to get a list of the medoids selected
print(Enter.tree.cluster)

#Re-clustering the tree using a threshold estimated by the GMMs but without the distribution
#cleaning (i.e. removing the overlapping and low count distributions)
Enter.tree.reclust <- ggrasp.recluster(Enter.tree.cluster, z.limit=0, min.lambda = 0)
```

`ggrasp.write` *ggrasp.write*

Description

writes formatted information from a class GGRaSP object to a file. Multiple output options are available.

Usage

```
ggrasp.write(x, file, type, rank.level)
```

Arguments

`x`	ggrasp-class object to be written
`file`	String pointing to file where the data will be saved. If no file is given, the result will be printed out on the screen.
`type`	Format of the data printed, either "tree" (New Hampshire extended style), "table" where the medoids or representative are shown in a table format, "list" where the information is shown in a pseudo-fasta format, or "itol" which prints out a file that can be loaded into the itol phylogeny viewer (http://itol.embl.de) which will color the clades of the different clusters
`rank.level`	Maximum level of the rank to show. Ignored pre-clustering. After clustering, 0 will show only the medoids, -1 will show all values independent of rank, and any value >= 1 will show all the genomes less than or equal to that rank (including medoids). Default is 0 (only the medoids)

Examples

```
#Getting the ggrasp object
Enter.tree <- ggrasp.load(system.file("extdata", "Enter.kSNP.tree", package="ggrasp"),
file.format = "tree", rank.file =system.file("extdata", "Enter.kSNP.ranks", package="ggrasp"));
Enter.tree.cluster <- ggrasp.cluster(Enter.tree)

#Default examples: using the initizalized ggrasp object will
#write the newick tree string to "tree.nwk"
```

```
ggrasp.write(Enter.tree, type="tree", file=file.path(tempdir(), "tree.nwk"));

# Using the clustered ggrasp object will write a text file with the clusters saved as an ITOL clade
# In conjecture with the phylogeny, this is readable by
# ITOL web phylogeny visualizer (http://itol.embl.de/)
ggrasp.write(Enter.tree.cluster, type="itol", file=file.path(tempdir(), "tree.itol.clade.txt"));
```

`plot.ggrasp` *plot.ggrasp*

Description

plots a class GGRaSP variable either the full tree, a reduced tree, or the gmm.

Usage

```
## S3 method for class 'ggrasp'
plot(x, type, layout = "circular", ...)
```

Arguments

`x`	ggrasp-class object to be plotted
`type`	Type of plot generated, either "tree" (the full tree with the clusters shown as grouped leaves), "reduced" (tree with only the medoids shown), "hist" (histogram of the distribution of the distances) or "gmm" (histogram of the distances overlayed with the Gaussian distributions)
`layout`	The layout style of the tree, either "circular" (default), "radial", "slanted", "linear" or "rectangular" ("linear" or "rectangular" are the same).
`...`	ignored

Value

A ggplot object containing the plot. It can be printed to standard output or saved using ggsave.

`print.ggrasp` *print.ggrasp*

Description

prints formatted information from a class GGRaSP object. Multiple output options are available.

Usage

```
## S3 method for class 'ggrasp'
print(x, type, rank.level, ...)
```

Arguments

x	ggrasp-class object to be printed
type	Format of the data printed, either "tree" (new hampshire extended style), "table" where the medoids or representative are shown in a table format, or "list" where the information is shown in a pseudo-fasta format
rank.level	Maximum level of the rank to show. Ignored pre-clustering. After clustering, 0 will show only the medoids, -1 will show all values independent of rank, and any value >= 1 will show all the genomes less than or equal to that rank (including medoids). Default is 0 (only the medoids)
...	ignored

Examples

```
#Getting the ggrasp object
Enter.tree <- ggrasp.load(system.file("extdata", "Enter.kSNP.tree", package="ggrasp"),
file.format = "tree", rank.file =system.file("extdata", "Enter.kSNP.ranks", package="ggrasp"));
Enter.tree.cluster <- ggrasp.cluster(Enter.tree)

#Default examples: using the initizalized ggrasp object will print the newick tree string
print(Enter.tree);

# Using the clustered ggrasp object will print the medoids and their respective clusters
print(Enter.tree.cluster)
#Below are examples of using different output formats and rank levels
print(Enter.tree.cluster, "tree")
print(Enter.tree.cluster, "table", 1)
print(Enter.tree.cluster, "table", 0)
```

---

summary.ggrasp *summary.ggrasp*

Description

prints a summary of the class GGRaSP variable. Output includes medoids and cutoff value after the clustering

Usage

```
## S3 method for class 'ggrasp'
summary(object, ...)
```

Arguments

object	ggrasp-class object
...	ignored

Index