Package: microSTASIS (via r-universe)

November 1, 2026

Title Microbiota STability ASsessment via Iterative cluStering

Version 1.12.0

Description The toolkit 'µSTASIS', or microSTASIS, has been developed for the stability analysis of microbiota in a temporal framework by leveraging on iterative clustering. Concretely, the core function uses Hartigan-Wong k-means algorithm as many times as possible for stressing out paired samples from the same individuals to test if they remain together for multiple numbers of clusters over a whole data set of individuals. Moreover, the package includes multiple functions to subset samples from paired times, validate the results or visualize the output.

License GPL-3

Imports BiocParallel, ggplot2, ggside, grid, rlang, stats, stringr, TreeSummarizedExperiment

Suggests BiocStyle, gghighlight, knitr, rmarkdown, methods, RefManageR, sessioninfo, SingleCellExperiment, SummarizedExperiment, testthat (>= 3.0.0)

biocViews GeneticVariability, BiomedicalInformatics, Clustering, MultipleComparison, Microbiome

BugReports https://github.com/BiotechPedro/microSTASIS

URL https://doi.org/10.1093/bib/bbac055

Encoding UTF-8

LazyData FALSE

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.1

VignetteBuilder knitr

Depends R (>= 4.2.0)

Config/testthat/edition 3

PackageStatus Deprecated

Config/pak/sysreqs cmake make libicu-dev libuv1-dev zlib1g-dev

Repository Bioconductor 3.23

Date/Publication 2026-04-28 12:59:28 UTC

RemoteUrl https://github.com/bioc/microSTASIS

RemoteRef RELEASE_3_23

RemoteSha 75388f6599b55755ba0fbb69cccf72390516f31f

Contents

clr . . . 2
iterativeClustering . . . 3
iterativeClusteringCV . . . 4
microSTASIS . . . 5
mSerrorCV . . . 5
mSinternalPairedTimes . . . 6
mSmetadataGroups . . . 7
mSpreviz . . . 8
pairedTimes . . . 8
plotmSdynamics . . . 10
plotmSheatmap . . . 11
plotmSlinesCV . . . 12
plotmSscatter . . . 13

Index **14**

`clr` *Detected ASV from multiple individuals at four different sampling times.*

Description

A dataset containing the amplicon sequence variants of 131 samples from the gut microbiota of 43 individuals. The values are transformed from counts by applying centred log-transformation (CLR).

Usage

```
data(clr)
```

Format

A data.frame with 131 rows and 226 variables

References

Gloria M. Agudelo-Ochoa, Beatriz E. Valdés-Duque, Nubia A. Giraldo-Giraldo, Ana M. Jaillier-Ramírez, Adriana Giraldo-Villa, Irene Acevedo-Castaño, Mónica A. Yepes-Molina, Janeth Barbosa-Barbosa, Alfonso Benítez-Paéz, Gut microbiota profiles in critically ill patients, potential biomarkers and risk variables for sepsis, Gut Microbes, Volume 12, Issue 1, January 2020, https://doi.org/10.1080/19490976.2019.170

Pedro Sánchez-Sánchez, Francisco J Santonja, Alfonso Benítez-Páez, Assessment of human microbiota stability across longitudinal samples using iteratively growing-partitioned clustering, Briefings in Bioinformatics, Volume 23, Issue 2, March 2022, bbac055, https://doi.org/10.1093/bib/bbac055

`iterativeClustering` *Stability of individuals after iteratively performing Hartigan-Wong k-means clustering.*

Description

Perform Hartigan-Wong `stats::kmeans()` algorithm as many times as possible. The values of k are from 2 to the number of samples minus 1. Those individuals whose paired samples are clustered under the same label sum 1. If paired samples are in different clusters, then sum 0, except when the euclidean distance between them is smaller to the ones of each sample to its centroid. This is done for all possible values of k and, finally, divided the sum by k, so obtaining a value between 0 and 1.

Usage

```
iterativeClustering(
  pairedTimes,
  BPPARAM = BiocParallel::bpparam(),
  common = "_"
)
```

Arguments

`pairedTimes`	list of matrices with paired times, i.e. samples to be stressed to multiple iterations. Output of `pairedTimes()`.
`BPPARAM`	supply a `BiocParallel` parameters object, e.g. `BiocParallel::SerialParam()` in the specific case of Windows OS or `BiocParallel::bpparam()`.
`common`	pattern that separates the ID and the sampling time.

Value

µSTASIS stability score (mS) for the individuals from the corresponding paired times.

Examples

```
data(clr)
times <- pairedTimes(data = clr, sequential = TRUE, common = "_0_")
mS <- iterativeClustering(pairedTimes = times, common = "_")
```

`iterativeClusteringCV` *Cross validation of the iterative Hartigan-Wong k-means clustering.*

Description

Perform cross validation of the stability results from `iterativeClustering()`in the way of leave-one-out (LOO) or leave-k-out (understood as quitting k individuals each time for calculating the metric over individuals).

Usage

```
iterativeClusteringCV(
  pairedTimes,
  results,
  name,
  common = "_",
  k = 1L,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

`pairedTimes`	list of matrices with paired times, i.e. samples to be stressed to multiple iterations. Output of `pairedTimes()`.
`results`	the list output of `iterativeClustering()`.
`name`	character; name of the paired times whose stability is being assessed.
`common`	pattern that separates the ID and the sampling time.
`k`	integer; number of individuals to remove from the data for each time running `iterativeClustering()`.
`BPPARAM`	supply a `BiocParallel` parameters object, e.g. `BiocParallel::SerialParam()` in the specific case of Windows OS or `BiocParallel::bpparam()`.

Value

Multiple lists with multiple objects of class "kmeans".

Examples

```
data(clr)
times <- pairedTimes(data = clr[, 1:20], sequential = TRUE, common = "_0_")
mS <- iterativeClustering(pairedTimes = times, common = "_")
cv_klist_t1_t25_k2 <- iterativeClusteringCV(pairedTimes = times,
                                            results = mS, name = "t1_t25",
                                            common = "_0_", k = 2L)
```

microSTASIS *Microbiota STability ASsessment via Iterative cluStering*

Description

The toolkit 'µSTASIS' has been developed for the stability analysis of microbiota in a temporal framework by leveraging on iterative clustering. Concretely, the core function uses Hartigan-Wong k-means algorithm as many times as possible for stressing out paired samples from the same individuals to test if they remain together for multiple numbers of clusters over a whole data set of individuals. Moreover, the package includes multiple functions to subset samples from paired times, validate the results or visualize the output.

mSerrorCV *Compute the mean absolute error (MAE) in percentage after* `iterativeClusteringCV()`.

Description

Compute the mean absolute error after the cross validation or plot lines connecting the stability values for each subset of the original matrix of paired times.

Usage

```
mSerrorCV(pairedTime, CVklist, k = 1L)
```

Arguments

`pairedTime`	input matrix with paired times whose stability has being assessed. One of the lists output of `pairedTimes()`.
`CVklist`	list resulting from `iterativeClusteringCV()`.
`k`	integer; number of individuals to subset from the data. The same as used in `iterativeClusteringCV()`.

Value

A vector with MAE values for each individual's mS score.

Examples

```
data(clr)
times <- pairedTimes(data = clr[, 1:20], sequential = TRUE, common = "_0_")
mS <- iterativeClustering(pairedTimes = times, common = "_")
cv_klist_t1_t25_k2 <- iterativeClusteringCV(pairedTimes = times,
                                            results = mS, name = "t1_t25",
                                            common = "_0_", k = 2L)
MAE_t1_t25 <- mSerrorCV(pairedTime = times$t1_t25,
```

```
                        CVklist = cv_klist_t1_t25_k2,  k = 2L)
MAE <- mSpreviz(results = list(MAE_t1_t25),
                times = list(t1_t25 = times$t1_t25))
plotmSheatmap(results = MAE, times = c("t1_t25", "t25_t26"), label = TRUE,
              high = 'red2',  low = 'forestgreen', midpoint = 5)
```

`mSinternalPairedTimes` *Internal function for* `pairedTimes()`.

Description

Internal function for `pairedTimes()`.

Usage

```
mSinternalPairedTimes(data, specifiedTimePoints, common = "_")
```

Arguments

`data`	matrix with rownames including ID, common pattern and sampling time.
`specifiedTimePoints`	character vector to specify the selection of concrete paired times.
`common`	pattern separating the ID and the sampling time in rownames.

Value

A list of matrices with the same number of columns as input and with samples from paired sampling times as rows.

Examples

```
data(clr)
t1_t2 <- mSinternalPairedTimes(data = clr,
                               specifiedTimePoints = c("1", "25"),
                               common = "_0_")
```

`mSmetadataGroups` *Easily extract groups of individuals from sample metadata.*

Description

Easily extract groups of individuals from sample metadata.

Usage

```
mSmetadataGroups(
  metadata,
  samples,
  individuals,
  variable,
  common,
  ID,
  timePoints
)
```

Arguments

`metadata`	input data.frame with data corresponding to samples. It can be the `SummarizedExperiment::colData()` from the TreeSummarizedExperiment.
`samples`	vector from metadata corresponding to the samples ID, if applicable; should be NULL if ID and timePoints are provided from a TreeSummarizedExperiment, for example.
`individuals`	vector of individuals; first column of the `mSpreviz()` output.
`variable`	column name with the variable used for grouping individuals.
`common`	pattern that separates the ID and the sampling time in rownames, if applicable.
`ID`	If applicable, one of the colData() colnames from the TreeSummarizedExperiment should be given as individuals.
`timePoints`	If applicable, one of the colData() colnames from the TreeSummarizedExperiment should be given as sampling times.

Value

A vector with the same length as the number of rows in the `mSpreviz()` output.

Examples

```
data(clr)
times <- pairedTimes(data = clr, sequential = TRUE, common = "_0_")
mS <- iterativeClustering(pairedTimes = times, common = "_")
results <- mSpreviz(results = mS, times = times)
metadata <- data.frame(Sample = rownames(clr), age = c(rep("youth", 65),
                       rep("old", 131-65)))
```

```
group <- mSmetadataGroups(metadata = metadata, samples = metadata$Sample,
                          common = "_0_", individuals = results$individual,
                          variable = "age")
```

`mSpreviz` *Process the* `iterativeClustering()` *output to a new format ready for the implemented visualization functions.*

Description

Process the `iterativeClustering()` output to a new format ready for the implemented visualization functions.

Usage

```
mSpreviz(results, times)
```

Arguments

`results`	list; output of `iterativeClustering()`.
`times`	list; output of `pairedTimes()`.

Value

A data frame ready for its use under the implemented visualization functions and others.

Examples

```
data(clr)
times <- pairedTimes(data = clr, sequential = TRUE, common = "_0_")
mS <- iterativeClustering(pairedTimes = times, common = "_")
results <- mSpreviz(results = mS, times = times)
```

`pairedTimes` *Generate one or multiple matrices with paired times.*

Description

Generate one or multiple matrices with paired times.

Usage

```
pairedTimes(data, ...)

## S4 method for signature 'matrix'
pairedTimes(data, sequential, common, specifiedTimePoints)

## S4 method for signature 'TreeSummarizedExperiment'
pairedTimes(
  data,
  sequential,
  assay,
  alternativeExp,
  ID,
  timePoints,
  specifiedTimePoints
)
```

Arguments

data	input object: either a matrix with rownames including ID, common pattern and sampling time, or a TreeSummarizedExperiment object.
...	Additional argument list that might not ever be used.
sequential	TRUE if paired times to analyse are sequential and present the desired alphanumerical order.
common	If is.matrix(data), pattern that separates the ID and the sampling time in rownames.
specifiedTimePoints	character vector to specify the selection of concrete paired times.
assay	If class(data) == "TreeSummarizedExperiment", name of the assay to use.
alternativeExp	If class(data) == "TreeSummarizedExperiment", name of the alternative experiment to use (if applicable).
ID	If class(data) == "TreeSummarizedExperiment", one of the colData(data) colnames should be given as individuals.
timePoints	If class(data) == "TreeSummarizedExperiment", one of the colData(data) colnames should be given as sampling times.

Value

A list of matrices with the same number of columns as input and with samples from paired sampling times as rows.

Examples

```
data(clr)
times <- pairedTimes(data = clr, sequential = TRUE, common = "_0_")
times_b <- pairedTimes(data = clr, sequential = FALSE, common = "_0_",
                       specifiedTimePoints = c("1", "26"))
```

`plotmSdynamics` *Generate boxplots of the stability dynamics throughout sampling times by groups.*

Description

Generate boxplots of the stability dynamics throughout sampling times by groups.

Usage

```
plotmSdynamics(results, groups, points = TRUE, linetype = 2)
```

Arguments

`results`	input data.frame resulting from `mSpreviz()`.
`groups`	vector with the same length as individuals, i.e. the number of rows in the `mSpreviz()` output.
`points`	logical; FALSE to only visualize boxplots or TRUE to also add individual points.
`linetype`	numeric; type of line to connect the median value of paired times; 0 to avoid the line.

Value

A plot with as many boxes as paired times by group in the form of a `ggplot2::ggplot()` object.

Examples

```
data(clr)
times <- pairedTimes(data = clr, sequential = TRUE, common = "_0_")
mS <- iterativeClustering(pairedTimes = times, common = "_")
results <- mSpreviz(results = mS, times = times)
metadata <- data.frame(Sample = rownames(clr), age = c(rep("youth", 65),
                       rep("old", 131-65)))
group <- mSmetadataGroups(metadata = metadata, samples = metadata$Sample,
                          common = "_0_", individuals = results$individual,
                          variable = "age")
plotmSdynamics(results, groups = group, points = TRUE, linetype = 0)
```

plotmSheatmap *Plot a heatmap of the stability results.*

Description

Plot a heatmap of the stability results.

Usage

```
plotmSheatmap(
  results,
  order = NULL,
  times,
  label = FALSE,
  low = "red2",
  mid = "yellow",
  high = "forestgreen",
  midpoint = 0.5
)
```

Arguments

results	input data.frame resulting from mSpreviz().
order	NULL object or character: none, mean or median; if the individuals should be sorted by any of those statistics of the stability values.
times	character; names of the paired times to plot, i.e. colnames of results.
label	logical; TRUE to print the mS score or FALSE to not.
low	color for the lowest value.
mid	color for the middle value.
high	color for the highest values.
midpoint	value to situate the middle.

Value

A heatmap of the stability values in the form of a ggplot2::ggplot() object.

Examples

```
data(clr)
times <- pairedTimes(data = clr, sequential = TRUE, common = "_0_")
mS <- iterativeClustering(pairedTimes = times, common = "_")
results <- mSpreviz(results = mS, times = times)
plotmSheatmap(results = results, order = "mean", times = c("t1_t25", "t25_t26"),
          label = TRUE)
```

plotmSlinesCV *Plot the stability values after* iterativeClusteringCV().

Description

Plot lines connecting the mS score for each subset of the original matrix of paired times.

Usage

```
plotmSlinesCV(pairedTime, CVklist, k = 1L, points = TRUE, sizeLine = 0.5)
```

Arguments

pairedTime	input matrix with paired times whose stability has being assessed. One of the lists output of pairedTimes().
CVklist	list resulting from iterativeClusteringCV().
k	integer; number of individuals to subset from the data. The same as used in iterativeClusteringCV().
points	logical; if plotting, FALSE to only plot lines and TRUE to add points on the mS score, i.e. result from iterativeClusteringCV().
sizeLine	numeric; if plotting, size of the multiple lines.

Value

A line plot in the form of a ggplot2::ggplot() object with the values of stability for the multiple subsets and the original matrix of paired samples (points).

Examples

```
data(clr)
times <- pairedTimes(data = clr[, 1:20], sequential = TRUE, common = "_0_")
mS <- iterativeClustering(pairedTimes = times, common = "_")
cv_klist_t1_t25_k2 <- iterativeClusteringCV(pairedTimes = times,
                                            results = mS, name = "t1_t25",
                                            common = "_0_", k = 2L)
plotmSlinesCV(pairedTime = times$t1_t25, CVklist = cv_klist_t1_t25_k2, k = 2L)
```

plotmSscatter *Plot a scatter and side boxplot of the stability results.*

Description

Plot a scatter and side boxplot of the stability results.

Usage

```
plotmSscatter(results, order = NULL, times, gridLines = FALSE, sideScale = 0.3)
```

Arguments

results	input data.frame resulting from mSpreviz().
order	NULL object or character: mean or median; if the individuals should be sorted by any of those statistics of the stability values.
times	a vector with the names of each paired time, e.g. "t1_t2".
gridLines	logical; FALSE to print a blank background or TRUE to include a gray grid.
sideScale	numeric; scale of the side boxplot.

Value

A scatter plot and a side boxplot of the stability values in the form of a ggplot2::ggplot() object.

Examples

```
data(clr)
times <- pairedTimes(data = clr, sequential = TRUE, common = "_0_")
mS <- iterativeClustering(pairedTimes = times, common = "_")
results <- mSpreviz(results = mS, times = times)
plotmSscatter(results = results, order = "median", times = c("t1_t25",
                                                             "t25_t26"),
          gridLines = TRUE, sideScale = 0.2)
```

Index

∗ **datasets**
clr, 2

BiocParallel::bpparam(), *3*, *4*
BiocParallel::SerialParam(), *3*, *4*

clr, 2

ggplot2::ggplot(), *10–13*

iterativeClustering, 3
iterativeClustering(), *4*, *8*
iterativeClusteringCV, 4
iterativeClusteringCV(), *5*, *12*

microSTASIS, 5
mSerrorCV, 5
mSinternalPairedTimes, 6
mSmetadataGroups, 7
mSpreviz, 8
mSpreviz(), *7*, *10, 11*, *13*

pairedTimes, 8
pairedTimes(), *3–6*, *8*, *12*
pairedTimes,matrix,matrix-method
(pairedTimes), 8
pairedTimes,matrix-method
(pairedTimes), 8
pairedTimes,TreeSummarizedExperiment,TreeSummarizedExperiment-method
(pairedTimes), 8
pairedTimes,TreeSummarizedExperiment-method
(pairedTimes), 8
plotmSdynamics, 10
plotmSheatmap, 11
plotmSlinesCV, 12
plotmSscatter, 13

stats::kmeans(), *3*
SummarizedExperiment::colData(), *7*