Package ‘miaTime’

August 23, 2025

Type Package

Title Microbiome Time Series Analysis

Version 0.99.9

Description The `miaTime` package provides tools for microbiome time series analysis based on (Tree)SummarizedExperiment infrastructure.

biocViews Microbiome, Software, Sequencing

License Artistic-2.0 | file LICENSE

Depends R (>= 4.5.0), mia

Imports dplyr, methods, S4Vectors, SingleCellExperiment, stats, SummarizedExperiment, tidyr, TreeSummarizedExperiment

Suggests BiocStyle, devtools, ggplot2, knitr, lubridate, miaViz, rmarkdown, scater, testthat, vegan

URL https://github.com/microbiome/miaTime

BugReports https://github.com/microbiome/miaTime/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/miaTime

git_branch devel

git_last_commit 911ef84

git_last_commit_date 2025-08-04

Repository Bioconductor 3.22

Date/Publication 2025-08-22

Author Leo Lahti [aut] (ORCID: <https://orcid.org/0000-0001-5537-637X>),
Tuomas Borman [aut, cre] (ORCID: <https://orcid.org/0000-0002-8563-8884>),
Yagmur Simsek [aut],
Sudarshan Shetty [ctb],
Chouaib Benchraka [ctb],
Muluh Muluh [ctb]

Maintainer Tuomas Borman <tuomas.v.borman@utu.fi>

Contents

`miaTime-package` `miaTime` *Package.*

Description

`miaTime` implements time series methods in mia ecosystem.

Author(s)

Maintainer: Tuomas Borman <`tuomas.v.borman@utu.fi`> (ORCID)

Authors:

- Leo Lahti <`leo.lahti@iki.fi`> (ORCID)
- Yagmur Simsek <`yagmur.simsek@hsrw.org`>

Other contributors:

- Sudarshan Shetty <`sudarshanshetty9@gmail.com`> [contributor]
- Chouaib Benchraka [contributor]
- Muluh Muluh [contributor]

See Also

TreeSummarizedExperiment class

addShortTermChange *Short term changes in abundance*

Description

Calculates short term changes in abundance of taxa using temporal abundance data.

Usage

```
addShortTermChange(x, ...)

getShortTermChange(x, ...)

## S4 method for signature 'SummarizedExperiment'
addShortTermChange(x, name = "short_term_change", ...)

## S4 method for signature 'SummarizedExperiment'
getShortTermChange(x, time.col, assay.type = "counts", group = NULL, ...)
```

Arguments

`x`	A `SummarizedExperiment` object.
`...`	additional arguments. • `time.interval`: `Integer scalar`. Indicates the increment between time steps. By default, the function compares each sample to the previous one. If you need to take every second, every third, or so, time step, then increase this accordingly. (Default: `1L`)
`name`	`Character scalar`. Specifies a name for storing short term results. (Default: `"short_term_change"`)
`time.col`	`Character scalar`. Specifies a name of the column from `colData` that identifies the sampling time points for the samples.
`assay.type`	`Character scalar`. Specifies which assay values are used in the dissimilarity estimation. (Default: `"counts"`)
`group`	`Character scalar`. Specifies a name of the column from `colData` that identifies the grouping of the samples. (Default: `NULL`)

Details

These functions can be utilized to calculate growth metrics for short term change. In specific, the functions calculate the metrics with the following equations:

$$time\_diff = time_t - time_{t-1}$$

$$abundance\_diff = abundance_t - abundance_{t-1}$$

$$growth\_rate = abundance\_diff - abundance_{t-1}$$

$$rate\_of\_change = abundance\_diff - time\_diff$$

Value

getShortTermChange returns DataFrame object containing the short term change in abundance over time for a microbe. addShortTermChange, on the other hand, returns a SummarizedExperiment object with these results in its metadata.

References

Ji, B.W., et al. (2020) Macroecological dynamics of gut microbiota. Nat Microbiol 5, 768–775 . doi: https://doi.org/10.1038/s41564-020-0685-1

See Also

getStepwiseDivergence()

Examples

```
library(miaTime)

# Load time series data
data(minimalgut)
tse <- minimalgut

# Get relative abundances
tse <- transformAssay(tse, method = "relabundance")
# Calculate short term changes
df <- getShortTermChange(
  tse, assay.type = "relabundance", time.col = "Time.hr",
  group = "StudyIdentifier")

# Calculate the logarithm of the ratio described in Ji, B.W., et al. (2020)
tse <- transformAssay(
    tse, assay.type = "relabundance", method = "log10", pseudocount = TRUE)
df <- getShortTermChange(
  tse, assay.type = "log10", time.col = "Time.hr", group = "StudyIdentifier")

# Select certain bacteria that have highest growth rate
select <- df[["growth_rate"]] |> abs() |> order(decreasing = FALSE)
select <- df[select, "FeatureID"] |> unique() |> head(3)
df <- df[ which(df[["FeatureID"]] %in% select), ]

# Plot results
library(ggplot2)
p <- ggplot(df, aes(x = Time.hr, y = growth_rate, colour = FeatureID)) +
    geom_smooth(level = 0.5) +
    facet_grid(. ~ StudyIdentifier, scales = "free") +
    scale_y_log10()
p
```

getBaselineDivergence *Beta diversity between the baseline and later time steps*

Description

Calculates sample dissimilarity between the given baseline and other time points, optionally within a group (subject, reaction chamber, or similar). The corresponding time difference is returned as well.

Usage

```
getBaselineDivergence(x, ...)

addBaselineDivergence(x, ...)

## S4 method for signature 'SummarizedExperiment'
getBaselineDivergence(
  x,
  time.col,
  assay.type = "counts",
  reference = NULL,
  group = NULL,
  method = "bray",
  ...
)

## S4 method for signature 'SummarizedExperiment'
addBaselineDivergence(
  x,
  name = c("divergence", "time_diff", "ref_samples"),
  ...
)
```

Arguments

`x`	A `SummarizedExperiment` object.
`...`	Optional arguments passed into `mia::addDivergence()`.
`time.col`	`Character scalar`. Specifies a name of the column from `colData` that identifies the sampling time points for the samples.
`assay.type`	`Character scalar`. Specifies which assay values are used in the dissimilarity estimation. (Default: `"counts"`)
`reference`	`Character scalar`. Specifies a name of the column from `colData` that identifies the baseline samples to be used. (Default: `NULL`)
`group`	`Character scalar`. Specifies a name of the column from `colData` that identifies the grouping of the samples. (Default: `NULL`)
`method`	`Character scalar`. Used to calculate the dissimilarity Method is passed to the function that is specified by `dis.fun`. (Default: `"bray"`)
`name`	`Character vector`. Specifies a column name for storing divergence results. (Default: `c("divergence", "time_diff", "ref_samples")`)

Details

The group argument allows calculating divergence per group. If given, the divergence is calculated per group. e.g. subject, chamber, group etc. Otherwise, this is done across all samples at once.

The baseline sample(s) always need to belong to the data object i.e. they can be merged into it before applying this function. The reason is that they need to have comparable sample data, at least some time point information for calculating time differences w.r.t. baseline sample.

The baseline time point is by default defined as the smallest time point (per group). Alternatively, the user can provide the baseline vector, or a list of baseline vectors per group (named list per group).

Value

getBaselineDivergence returns DataFrame object containing the sample dissimilarity and corresponding time difference between samples. addBaselineDivergence, on the other hand, returns a SummarizedExperiment object with these results in its colData.

See Also

mia::addDivergence()

Examples

```
library(miaTime)

data(hitchip1006)
tse <- transformAssay(hitchip1006, method = "relabundance")

# By default, reference samples are the samples from the first timepoint
tse <- addBaselineDivergence(
    tse,
    group = "subject",
    time.col = "time",
    assay.type = "relabundance",
    method = "bray")

# Add reference samples to colData, if you want to specify reference
# samples manually
colData(tse)[["reference"]] <- "Sample-875"
tse <- addBaselineDivergence(
    tse,
    reference = "reference",
    group = "subject",
    time.col = "time",
    name = c("divergence_from_baseline",
        "time_from_baseline", "reference_samples"),
    assay.type = "relabundance",
    method = "bray")
```

getBimodality *Calculate coefficient of bimodality.*

Description

This function calculates coefficient of bimodality for each taxa.

Usage

```
getBimodality(x, ...)

addBimodality(x, ...)

## S4 method for signature 'SummarizedExperiment'
addBimodality(x, name = "bimodality", ...)

## S4 method for signature 'SummarizedExperiment'
getBimodality(x, assay.type = "counts", ...)
```

Arguments

`x`	A `SummarizedExperiment` object.
`...`	additional arguments. • `group`: `Character scalar`. Specifies a name of the column from `colData` that identifies the grouping of the samples. (Default: `NULL`)
`name`	`Character scalar`. Specifies a column name for storing bimodality results. (Default: `"bimodality"`)
`assay.type`	`Character scalar`. Specifies which assay values are used in the dissimilarity estimation. (Default: `"counts"`)

Details

This function calculates coefficient of bimodality for each taxa. If the dataset includes grouping, for instance, individual systems or patients, the coefficient is calculated for each group separately.

The coefficient of bimodality measures whether a taxon has bimodal abundance. For instance, certain taxon can be high-abundant in some timepoints while in others it might be rare. The coefficient can help to determine these taxa.

The coefficient of bimodality (b) is defined as follows:

$$b = \frac{1 + skewness^2}{kurtosis + 3}$$

where skewness is calculated as follows

$$skewness = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^3/n}{\sum_{i=1}^{n}(x_i - \overline{x})^2/n]^{3/2}}$$

and kurtosis as follows

$$kurtosis = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^4/n}{(\sum_{i=1}^{n}(x_i - \overline{x})^2/n)^2}$$

The coefficient ranges from 0-1, where 1 means high bimodality. The coefficient was introduced in the paper Shade A et al. 2014, where they used bimodality and abundance to determine conditionally rare taxa (CRT).

Value

`DataFrame` or `x` with results added to its `rowData`.

References

Shade A, et al. (2014) Conditionally Rare Taxa Disproportionately Contribute to Temporal Changes in Microbial Diversity. doi: 10.1128/mbio.01371-14

See Also

`mia::getConditionallyLowAbundant()`

Examples

```
library(miaTime)

data(SilvermanAGutData)
tse <- SilvermanAGutData

# In this example, we are only interested on vessel 1.
tse <- tse[, tse[["Vessel"]] == 1]
tse <- transformAssay(tse, method = "relabundance")

# Calculate bimodality
b <- getBimodality(tse)
# Determine taxa with high bimodality
bimodal_taxa <- names(b)[ which(b[[1]] > 0.95) ]

# Determine taxa with abundance > 0.5%
abundant <- getAbundant(
    tse, assay.type = "relabundance", abundant.th = 0.5/100)

# The detected CRT
crt <- intersect(bimodal_taxa, abundant)
head(crt)
```

`getStability` *Estimate stability*

Description

Quantify intermediate stability with respect to a given reference point.

Usage

```
getStability(x, ...)

addStability(x, ...)

## S4 method for signature 'SummarizedExperiment'
addStability(x, time.col, name = "stability", ...)

## S4 method for signature 'SummarizedExperiment'
getStability(
  x,
  time.col,
  assay.type = "counts",
  reference = NULL,
  group = NULL,
  ...
)
```

Arguments

`x` A `SummarizedExperiment` object.

`...` additional arguments.

- `time.interval`: `Integer scalar`. Indicates the increment between time steps. By default, the function compares each sample to the previous one. If you need to take every second, every third, or so, time step, then increase this accordingly. (Default: `1L`)
- `calc.separately`: `Logical scalar`. Specifies whether to calculate stability separately for data points with abundance below or at/above the reference point. (Defaul: `FALSE`)
- `mode`: `Character scalar`. Specifies whether to calculate stability using correlation ("correlation") or a linear model ("lm"). (Default: `"correlation"`)

`time.col` `Character scalar`. Specifies a name of the column from `colData` that identifies the sampling time points for the samples.

`name` `Character scalar`. Specifies a column name for storing stability results. (Default: `"stability"`)

`assay.type` `Character scalar`. Specifies which assay values are used in the dissimilarity estimation. (Default: `"counts"`)

`reference` `Character scalar`. Specifies a column from `rowData(x)` that determines the reference points. If `NULL`, the default reference is used. (Default: `NULL`)

`group` `Character scalar`. Specifies a name of the column from `colData` that identifies the grouping of the samples. (Default: `NULL`)

Details

These methods estimate intermediate stability described in Lahti et al. 2014. The method is heuristic and makes many simplifying assumptions. However, the stability estimation can be useful for exploration, and can provide a baseline for more advanced measures.

The stability is calculated by first defining reference point, R_f, for each feature. User can define the reference points with `reference`. If reference points are not defined, they are calculated by taking median

$$R_f = median(x_f)$$

where f denotes a single feature and x abundance.

Then difference between consecutive time points, $\Delta_{f,x}$,

$$\Delta_{f,x} = |x_{f,t} - x_{f,t-1}|$$

and difference between the previous time point and reference, $\Delta_{f.R}$, are calculated:

$$\Delta_{f.R} = |x_{f,t-1} - R_f|$$

where t denotes time point.

Optionally, time difference $\Delta_{f,t}$ is calculated

$$\Delta_{f,t} = t_{f,t-1} - t_{f,t-1}$$

The stability coefficient s_f is calculated with correlation

$$s_f = corr(\Delta_{f,x}, \Delta_{f,R})$$

or with linear model as follows

$$s_f = \frac{\Delta_{f,x} - \beta_0 - \beta_2 * \Delta_{f,t} - \epsilon_f}{\Delta_{f,R}}$$

Value

`getStability` returns `DataFrame` while `addStability` returns results added to its `rowData(x)`.

References

Lahti L, et al. (2014) Tipping elements in the human intestinal ecosystem. Nat Commun. doi: 10.1038/ncomms5344

See Also

`getBimodality()`

Examples

```
library(miaTime)

# Load time series data
data(minimalgut)
tse <- minimalgut

# Apply clr transformation
tse <- transformAssay(tse, method = "rclr")

# Calculate stability single system
tse_sub <- tse[, tse[["StudyIdentifier"]] == "Bioreactor A"]
tse_sub <- addStability(tse_sub, assay.type = "rclr", time.col = "Time.hr")
rowData(tse_sub)

# Add custom reference values
rowData(tse)[["ref_col"]] <- 1
# Calculate stability for each system simultaneously by taking time
# difference into account
tse <- addStability(
    tse, assay.type = "rclr", time.col = "Time.hr",
    group = "StudyIdentifier", ref_col = "ref_col", mode = "lm")
rowData(tse)
```

getStepwiseDivergence *Beta diversity between consecutive time steps*

Description

Calculates sample dissimilarity between consecutive time points along with time difference.

Usage

```
getStepwiseDivergence(x, ...)

addStepwiseDivergence(x, ...)

## S4 method for signature 'ANY'
getStepwiseDivergence(
  x,
  time.col,
  assay.type = "counts",
  time.interval = 1L,
  group = NULL,
  method = "bray",
  ...
)

## S4 method for signature 'SummarizedExperiment'
addStepwiseDivergence(
  x,
  name = c("divergence", "time_diff", "ref_samples"),
  ...
)
```

Arguments

x
: A SummarizedExperiment object.

...
: Optional arguments passed into mia::addDivergence().

time.col
: Character scalar. Specifies a name of the column from colData that identifies the sampling time points for the samples.

assay.type
: Character scalar. Specifies which assay values are used in the dissimilarity estimation. (Default: "counts")

time.interval
: Integer scalar. Indicates the increment between time steps. By default, the function compares each sample to the previous one. If you need to take every second, every third, or so, time step, then increase this accordingly. (Default: 1L)

group
: Character scalar. Specifies a name of the column from colData that identifies the grouping of the samples. (Default: NULL)

method
: Character scalar. Used to calculate the dissimilarity Method is passed to the function that is specified by dis.fun. (Default: "bray")

name
: Character vector. Specifies a column name for storing divergence results. (Default: c("divergence", "time_diff", "ref_samples"))

Details

These functions calculate time-wise divergence, meaning each sample is compared to the previous i-th sample, where i is the specified time interval (see `time.interval`). By default, the function calculates divergence by comparing all samples with each other. However, it is often more meaningful to calculate divergence within a specific patient or group (see the `group` parameter).

Value

`getStepwiseDivergence` returns `DataFrame` object containing the sample dissimilarity and corresponding time difference between samples. `addStepwiseDivergence`, on the other hand, returns a `SummarizedExperiment` object with these results in its `colData`.

See Also

`mia::addDivergence()`

Examples

```
library(miaTime)

data(hitchip1006)
tse <- transformAssay(hitchip1006, method = "relabundance")

# Calculate divergence
tse <- addStepwiseDivergence(
    tse,
    group = "subject",
    time.interval = 1,
    time.col = "time",
    assay.type = "relabundance"
    )
```

---

hitchip1006 *HITChip Atlas with 1006 Western Adults*

Description

This data set contains genus-level microbiota profiling with HITChip for 1006 western adults with no reported health complications, reported in Lahti et al. (2014).

Usage

```
data(hitchip1006)
```

Format

The data set in `TreeSummarizedExperiment` format.

Details

The data is also available for download from the Data Dryad `http://doi.org/10.5061/dryad.pk75d` and was initially released as a phyloseq object under the name atlas1006 in **microbiome** R package. The **miaTime** package provides this as `TreeSummarizedExperiment` object. Some of the subjects also have short time series.

Value

Loads the data set in R.

Author(s)

Leo Lahti

References

Lahti et al. Tipping elements of the human intestinal ecosystem. Nature Communications 5:4344, 2014. `https://doi.org/10.1038/ncomms5344`.

`minimalgut` *Human Gut Minimal Microbiome Profiling Data*

Description

This time-series data set contains absolute temporal abundances for 16 human gut species that were assembled in an in vitro gut system. These were subjected to a variety of disturbances over a period of 20 days. The sample data includes measurements for Acetate, Butyrate, Propionate, and optical density.

Usage

```
data(minimalgut)
```

Format

The data set in `TreeSummarizedExperiment` format.

Details

The data is available also on `https://github.com/microsud/syncomR` The data contains 183 samples from 3 in vitro gut system, 61 per bioreactor collected three times daily. The current implementation in **miaTime** provides this as `TreeSummarizedExperiment` object.

Value

Loads the data set in R.

Author(s)

Sudarshan A. Shetty

References

Shetty, S.A., Kostopoulos, I., Geerlings, S.Y. et al. Dynamic metabolic interactions and trophic roles of human gut microbes identified using a minimal microbiome exhibiting ecological properties. ISME, 16, 2144–2159 (2022). https://doi.org/10.1038/s41396-022-01255-2.

`SilvermanAGutData` *SilvermanAGutData*

Description

The SilvermanAGutData dataset contains 16S rRNA gene based high-throughput profiling of 4 in vitro artificial gut models. The sampling was done hourly and daily to capture sub-daily dynamics of microbial community originating from human feces. The data consists of 413 taxa from 639 samples. The data set can be used to investigate longitudinal dynamics of microbial community in a controlled environment.

Column metadata includes the days of sampling, vessel identifier, sampling frequency pre-post challenge with Bacteroides ovatus.

The row metadata of the microbiome data contains taxonomic information on the Kingdom, Phylum, Class, Order, Family and Genus and Species level.

The row tree consists of a phylogenetic tree build using sequence information of 413 taxa.

As reference sequences the ASV are provided.

All data are downloaded from ExperimentHub and cached for local re-use

Usage

```
data(SilvermanAGutData)
```

Format

The data set in `TreeSummarizedExperiment` format.

Value

Loads the data set in R.

Author(s)

Sudarshan A. Shetty and Felix G.M. Ernst

References

Silveman J.D et al. (2018): Dynamic linear models guide design and analysis of microbiota studies within artificial human guts. Microbiome 6:202 `https://doi.org/10.1186/s40168-018-0584-3`

temporalMicrobiome20 *Gut Microbiome Profiling 20 Belgian Women*

Description

This time-series data set contains absolute temporal variation for most major gut genera, as well as diversity indicators for both relative and quantitative abundance profiles of healthy women living in an industrialized country.

Usage

```
data(temporalMicrobiome20)
```

Format

The data set in `TreeSummarizedExperiment` format.

Details

The data is available also on `https://www.nature.com/articles/s41467-021-27098-7#Sec43` The data contains 713 fecal samples from 20 Belgian women collected over six weeks. The current implementation in **miaTime** provides this as `TreeSummarizedExperiment` object.

Value

Loads the data set in R.

Author(s)

Doris Vandeputte

References

Vandeputte, D., De Commer, L., Tito, R.Y. et al. Temporal variability in quantitative human gut microbiome profiles and implications for clinical research. Nat Commun 12, 6740 (2021). https://doi.org/10.1038/s41467-021-27098-7

Index