Package ‘ALDEx2’

March 22, 2017

Type Package
Title Analysis Of Differential Abundance Taking Sample Variation Into Account
Version 1.6.0
Date 2016-07-27
Author Greg Gloor, Ruth Grace Wong, Andrew Fernandes, Arianne Albert, Matt Links, Jia Rong Wu
Maintainer Greg Gloor <ggloor@uwo.ca>
Depends methods
Imports S4Vectors, IRanges, GenomicRanges, SummarizedExperiment, BiocParallel
biocViews DifferentialExpression, RNASeq, DNASeq, ChIPSeq, GeneExpression, Bayesian, Sequencing, Software, Microbiome, Metagenomics
Description A differential abundance analysis for the comparison of two or more conditions. For example, single-organism and meta-RNA-seq high-throughput sequencing assays, or of selected and unselected values from in-vitro sequence selections. Uses a Dirichlet-multinomial model to infer abundance from counts, that has been optimized for three or more experimental replicates. Infers sampling variation and calculates the expected false discovery rate given the biological and sampling variation using the Wilcox rank test or Welch’s t-test (aldex.ttest) or the glm and Kruskal Wallis tests (aldex.glm). Reports both P and fdr values calculated by the Benjamini Hochberg correction.
License file LICENSE
NeedsCompilation no

R topics documented:

ALDEx2m-package .................................................... 2
aldex ................................................................. 3
aldex.clr ............................................................ 4
aldex.clr-class ....................................................... 6
aldex.cor ............................................................ 8
aldex.effect .......................................................... 9
aldex.glm .......................................................... 10
ALDEx2m-package

Description

A differential abundance analysis for the comparison of two or more conditions. For example, single-organism and meta-RNA-seq high-throughput sequencing assays, or of selected and unselected values from in-vitro sequence selections. Uses a Dirichlet-multinomial model to infer abundance from counts, that has been optimized for three or more experimental replicates. Infers sampling variation and calculates the expected false discovery rate given the biological and sampling variation using the Wilcoxon rank test or Welch's t-test (aldex.ttest) or the glm and Kruskal Wallis tests (aldex.glm). Reports both P and fdr values calculated by the Benjamini Hochberg correction.

References

Please use the citation given by `citation(package="ALDEx")`.

See Also

`aldex.clr, aldex.ttest, aldex.glm, aldex.effect, selex`

Examples

```r
# see examples for the aldex.clr, aldex.ttest, aldex.effect, aldex.glm functions
```
Compute an aldex Object

Description

Generate Monte Carlo samples of the Dirichlet distribution for each sample. Convert each instance using the centred log-ratio transform. Return two sample test values (Welch's t, Wilcoxon) or multi-sample test values (glm or Kruskal Wallace). Returns effect size values by default.

Usage

```r
aldex(reads, conditions, mc.samples=128, test="t",
    effect=TRUE, include.sample.summary=FALSE, verbose=FALSE, denom="all")
```

Arguments

- `reads`: a non-negative, integer-only containing data.frame that has unique names for all rows and columns, where each row is a different gene and each column represents a sequencing read-count. Rows with 0 reads in each sample are deleted prior to analysis.
- `conditions`: a description of the data structure to be used for testing.
- `mc.samples`: the number of Monte Carlo samples to use to estimate the underlying distributions; since we are estimating central tendencies, 128 is usually sufficient.
- `denom`: A character variable default "all" indicating which features to retain as the denominator for the Geometric Mean calculation. Using "iqlr" accounts for data with systematic variation and centers the features on the set features that have variance that is between the lower and upper quartile of variance. Using "zero" is a more extreme case where there are many nonzero features in one condition but many zeros in another. In this case the geometric mean of each group is calculated using the set of per-group non-zero features.
- `test`: which tests to perform: t = Welch's t and Wilcoxon, glm = Kruskal Wallace and glm
- `effect`: calculate abundances and effect sizes
- `include.sample.summary`: include median clr values for each sample, defaults to FALSE
- `verbose`: Print diagnostic information while running. Useful only for debugging if fails on large datasets

Details

An explicit description of the input format for the `reads` object is shown under 'Examples', below. This is not intended to be the generic function. The system is intended to be used for demonstration or instructional purposes.

Value

returns a number of values that depends on the set of options. See the return values of `aldex.ttest`, `aldex glm`, and `aldex.effect` for explanations and example.
Author(s)

Greg Gloor, Andrew Fernandes and Matt Links contributed to this code.

References

Please use the citation given by citation(package="ALDEx").

See Also

aldex.ttest, aldex.glm, aldex.effect, aldex.corr, selex

Examples

# The 'reads' data.frame should have row
# and column names that are unique, and
# looks like the following:
#
# T1a T1b T2 T3 N1 N2 Nx
# Gene_00001 0 0 2 0 0 1 0
# Gene_00002 20 8 12 5 19 26 14
# Gene_00003 3 0 2 0 0 0 1
# Gene_00004 75 84 241 149 271 257 188
# Gene_00005 10 16 4 0 4 10 10
# Gene_00006 129 126 451 223 243 149 209
# ... many more rows ...

data(selex)
#subset for efficiency
selex <- selex[1201:1600,]
conds <- c(rep("NS", 7), rep("S", 7))
x <- aldex(selex, conds, mc.samples=2, denom="all", test="t",
effect=FALSE, verbose=FALSE)
conds  A vector containing a descriptor for the samples, allowing them to be grouped and compared.

mc.samples  The number of Monte Carlo samples to use to estimate the underlying distributions; since we are estimating central tendencies, 128 is usually sufficient.

denom  A character variable default "all" indicating which features to use as the denominator for the Geometric Mean calculation. Using "all" uses the geometric mean abundance of all features. Using "iqlr" accounts for data with systematic variation and uses the features that are between the first and third quartile of the variance of the clr values across all sample. Using "zero" uses the non-zero features in each groip as the denominator. This approach is an extreme case where there are many nonzero features in one condition but many zeros in another. It is also possible to supply a vector of row indices to use as the denominator. Here, the experimentalist is determining a-priori which rows are thought to be invariant. In the case of RNA-seq, this could include ribosomial protein genes and and other house-keeping genes.

verbose  Print diagnostic information while running. Useful only for debugging if fails on large datasets.

useMC  Use multicore by default (FALSE). Multi core processing will be attempted with the BiocParallel package. Serial processing will be used if this is not possible.

Details
An explicit description of the input format for the reads object is shown under ‘Examples’, below.

Value
The object produced by the clr function contains the clr transformed values for each Monte-Carlo Dirichlet instance, which can be accessed through getMonteCarloInstances(x), where x is the clr function output. Each list element is named by the sample ID. getFeatures(x) returns the features, getSampleIDs(x) returns sample IDs, and getFeatureNames(x) returns the feature names.

Author(s)
Greg Gloor, Ruth Grace Wong, Andrew Fernandes, Matt Links and Jia Rong Wu contributed to this code.

References
Please use the citation given by citation(package="ALDEx").

See Also
aldex.ttest, aldex.glm, aldex.effect, selex

Examples

# The 'reads' data.frame or # RangedSummarizedExperiment object should # have row and column names that are unique, # and looks like the following:
#
#          T1a T1b T2 T3 N1 N2 Nx

aldex.clr
The `aldex.clr-class` is an S4 class which stores the data generated by the `aldex.clr` method.

### Details

An `aldex.clr` object contains the Monte Carlo Dirichlet instances derived from estimating the technical variance of the raw read count data. It is created by the `aldex.clr.function`, which is invoked by the `aldex.clr` method. It consists of four attributes: the sample names, the feature names, the conditions vector (assigns each sample to a condition), and the Monte Carlo Dirichlet instances themselves. These can be accessed, along with information about the length of some attributes. A single Monte Carlo instance can also be retrieved.

### Value

The `aldex.clr` object contains the clr transformed values for each Monte-Carlo Dirichlet instance, which can be accessed through `getMonteCarloInstances(x)`, where `x` is the `clr` function output. Each list element is named by the sample ID. `getFeatures(x)` returns the features, `getSampleIDs(x)` returns sample IDs, and `getFeatureNames(x)` returns the feature names.

### Methods

In the code below, `x` is an `aldex.clr` object, and `i` is a numeric whole number.

- `getMonteCarloInstances(x)`: Returns `x`'s Monte Carlo Dirichlet instances.
- `getSampleIDs(x)`: Returns the names of the samples. These can be used to access the original reads, as in `reads$sampleID` (if the reads are a data frame).
- `getFeatures(x)`: Returns the names of the features as a vector.
- `numFeatures(x)`: Returns the number of features associated with the data.
- `numMCInstances(x)`: Returns the names of the keys that can be used to subset the data rows. The keys values are the rsid’s.
- `getFeatureNames(x)`: Returns the names of the keys that can be used to subset the data rows. The keys values are the rsid’s.
getReads(x): Returns the names of the keys that can be used to subset the data rows. The keys values are the rsid’s.

numConditions(x): Returns the names of the keys that can be used to subset the data rows. The keys values are the rsid’s.

getMonteCarloReplicate(x, i): Returns the names of the keys that can be used to subset the data rows. The keys values are the rsid’s.

Author(s)

Greg Gloor, Ruth Grace Wong, Andrew Fernandes, Jia Rong Wu and Matt Links contributed to this code

References

Please use the citation given by citation(package="ALDEx").

See Also

aldex.clr.function

Examples

# The 'reads' data.frame or
# SummarizedExperiment object should have
# row and column names that are unique,
# and looks like the following:
#
#    T1a T1b T2 T3 N1 N2 Nx
# Gene_00001  0  0  2  0  0  1  0
# Gene_00002  20  8 12  5 19 26 14
# Gene_00003   3  0  2  0  0  0  1
# Gene_00004   75  84 241 149 271 257 188
# Gene_00005   10 16  4  0  4 10 10
# Gene_00006 129 126 451 223 243 149 209
#    ... many more rows ...

data(selex)
#subset for efficiency
selex <- selex[1201:1600,]
conds <- c(rep("NS", 7), rep("S", 7))

# x is an object of type aldex.clr
x <- aldex.clr(selex, conds, mc.samples = 2, denom="all", verbose = FALSE)

# get all of the Monte Carlo Dirochlet instances
monteCarloInstances <- getMonteCarloInstances(x)

# get sample names
sampleIDs <- getSampleIDs(x)

# get features
features <- getFeatures(x)

# get number of features
numFeatures <- numFeatures(x)
# get number of Monte Carlo Dirichlet instances
numInstances <- numMCInstances(x)

# get names of features
featureNames <- getFeatureNames(x)

# get number of conditions
conditions <- numConditions(x)

# get number of conditions
reads <- getReads(x)

# retrieve the first Monte Carlo Dirichlet instance.
monteCarloInstance <- getMonteCarloReplicate(x,1)

aldex.corr  
\[\text{calculate Pearson's Product moment and Spearman's rank correlations}\]

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>calculates expected values of Pearson's Product moment and Spearman's rank correlations on the data returned by aldex.clr. NOTE: this function will be replaced by a compositionally correct method in the next release cycle.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>aldex.corr(clr, covar)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>clr</td>
</tr>
<tr>
<td>covar</td>
</tr>
</tbody>
</table>

| clr | clr is the data output of the aldex.clr function |
| covar | a per-sample continuous variable to be correlated with the clr values |

<table>
<thead>
<tr>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>An explicit example for two conditions is shown in the Examples below.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outputs a dataframe with the following information:</td>
</tr>
</tbody>
</table>

| pearson.eCor | a vector containing the expected Pearson's Product moment value for each feature |
| pearson.ep | a vector containing the expected P value of the Pearson Product moment value for each feature |
| pearson.eBH | a vector containing the expected Benjamini-Hochberg corrected P value of the Pearson Product moment value for each feature |
| spearman.erho | a vector containing the expected Spearman's rank correlation value for each feature |
**spearman.ep**  a vector containing the expected P value of Spearman’s rank correlation value for each feature

**spearman.eBH**  a vector containing the expected Benjamini-Hochberg corrected P value of Spearman’s rank correlation value for each feature

**Author(s)**

Arianne Albert

**References**

Please use the citation given by `citation(package="ALDEx")`.

**See Also**

`aldex.clr`, `aldex.glm`, `aldex.effect`, `selex`

**Examples**

```r
# x is the output of the `x <- aldex.clr(data, conds, mc.samples, denom="all", useMC)` function
# conditions is a description of the data
# aldex.ttest(clr, covar)

aldex.effect
```

---

**Description**

determines the median clr abundance of the feature in all samples and in groups determines the median difference between the two groups determines the median variation within each two group determines the effect size, which is the median of the ratio of the between group difference and the larger of the variance within groups

**Usage**

```r
aldex.effect(clr, conditions, verbose = TRUE, include.sample.summary = FALSE, useMC=FALSE)
```

**Arguments**

- **clr**  clr is the data output of `aldex.clr`
- **conditions**  a description of the data structure to be used for testing
- **verbose**  Print diagnostic information while running. Useful only for debugging if fails on large datasets
- **include.sample.summary**  include median clr values for each sample, defaults to FALSE
- **useMC**  use multicore by default (FALSE)

**Details**

An explicit example for two conditions is shown in the ‘Examples’ below.
aldex.glm

Value

returns a dataframe with the following information:

- **rab.all**: a vector containing the median clr value for each feature
- **rab.win.conditionA**: a vector containing the median clr value for each feature in condition A
- **rab.win.conditionB**: a vector containing the median clr value for each feature in condition B
- **diff.btw**: a vector containing the per-feature median difference between condition A and B
- **diff.win**: a vector containing the per-feature maximum median difference between Dirichlet instances within conditions
- **effect**: a vector containing the per-feature effect size
- **overlap**: a vector containing the per-feature proportion of effect size that is 0 or less

Author(s)

Greg Gloor, Andrew Fernandes, Matt Links

References

Please use the citation given by `citation(package="ALDEx")`.

See Also

`aldex.clr, aldex.ttest, aldex.glm, selex`

Examples

```r
# x is the output of the `x <- clr(data, mc.samples)` function
# conditions is a description of the data
# for the selex dataset, conditions <- c(rep("N", 7), rep("S", 7))
data(selex)
data <- selex[1201:1600,]
conds <- c(rep("NS", 7), rep("S", 7))
x <- aldex.clr(selex, conds, mc.samples=2, denom="all")
ext <- aldex.effect(x, conds)
```
Arguments

- clr: clr is the data output of aldex.clr
- conditions: a description of the data structure to be used for testing
- useMC: use multicore by default (FALSE)

Details

An explicit example for two conditions is shown in the ‘Examples’ below.

Value

Outputs a dataframe with the following information:

- kw.ep: a vector containing the expected P value of the Kruskal Wallis test for each feature
- kw.eBH: a vector containing the expected value of the Benjamini Hochberg corrected P value for each feature
- glm.ep: a vector containing the expected P value of the glm test for each feature
- glm.eBH: a vector containing the expected value of the Benjamini Hochberg corrected P value for each feature

Author(s)

Arianne Albert

References

Please use the citation given by citation(package="ALDEx").

See Also

aldex.clr, aldex.ttest, aldex.effect, selex

Examples

```r
# x is the output of the 
# code(x <- aldex.clr(data, mc.samples)) function
# conditions is a description of the data
# for the selex dataset, conditions <- c(rep("N", 7), rep("S", 7))
data(selex)
#subset for efficiency
selex <- selex[1201:1600,]
conds <- c(rep("NS", 7), rep("S", 7))
x <- aldex.clr(selex, conds, mc.samples=1, denom="all")
glm.test <- aldex.glm(x, conds)
```
Plot an aldex Object

Description

Create ‘MW’- or ‘MA’-type plots from the given aldex object.

Usage

```r
## S3 method for class 'plot'
aldex( x, ..., type=c("MW","MA"),
xlab=NULL, ylab=NULL, xlim=NULL, ylim=NULL,
all.col=rgb(0,0,0,0.2), all.pch=19, all.cex=0.4,
called.col=red, called.pch=20, called.cex=0.6,
thres.line.col=darkgrey, thres.lwd=1.5,
test=welch, cutoff=0.1, rare.col=black, rare=0,
rare.pch=20, rare.cex=0.2 )
```

Arguments

- `x`: an object of class aldex produced by the aldex function
- `...`: optional, unused arguments included for compatibility with the S3 method signature
- `type`: which type of plot is to be produced. MA is a Bland-Altman style plot; MW is a difference between to a variance within plot as described in the paper
- `test`: the method of calculating significance, one of: `welch` = welch’s t test; `wilcox` = wilcox rank test; `glm` = glm; `kruskal` = Kruskal-Wallace test
- `cutoff`: the Benjamini-Hochberg fdr cutoff, default 0.1
- `xlab`: the x-label for the plot, as per the parent `plot` function
- `ylab`: the y-label for the plot, as per the parent `plot` function
- `xlim`: the x-limits for the plot, as per the parent `plot` function
- `ylim`: the y-limits for the plot, as per the parent `plot` function
- `all.col`: the default colour of the plotted points
- `all.pch`: the default plotting symbol
- `all.cex`: the default symbol size
- `called.col`: the colour of points with false discovery rate, q <= 0.1
- `called.pch`: the symbol of points with false discovery rate, q <= 0.1
- `called.cex`: the character expansion of points with false discovery rate, q <= 0.1
- `thres.line.col`: the colour of the threshold line where within and between group variation is equivalent
- `thres.lwd`: the width of the threshold line where within and between group variation is equivalent
- `rare`: relative abundance cutoff for rare features, default 0 or the mean abundance
- `rare.col`: color for rare features, default black
- `rare.pch`: the default symbol of rare features
- `rare.cex`: the default symbol size of rare points

Notes

- Use `type="MA"` to produce a Bland-Altman plot.
- Use `type="MW"` to produce a plot as described in the paper.
- The `test` argument allows for different methods of calculating significance.
- The `cutoff` argument controls the Benjamini-Hochberg false discovery rate.
- The `xlab` and `ylab` arguments allow for custom labels.
- The `xlim` and `ylim` arguments set the plot limits.
- The `all.col`, `all.pch`, and `all.cex` arguments control the appearance of all points.
- The `called.col`, `called.pch`, and `called.cex` arguments control the appearance of points with a false discovery rate of 0.1.
- The `thres.line.col` and `thres.lwd` arguments control the appearance of the threshold line.
- The `rare` argument allows for the inclusion of rare features based on their relative abundance.
- The `rare.col`, `rare.pch`, and `rare.cex` arguments control the appearance of rare features.

Examples

```r
# Load data and create an aldex object
data <- read.table("data.txt", header=TRUE)
aldex_obj <- aldex(data)

# Plot using MW type
aldex.plot(aldex_obj, type="MW")
```

References

- Welch, B. L. (1947). The generalization of ‘Student’s’ problem when several different population variances are involved. Biometrika, 34(1-2), 28-35.
- glm: Generalized linear models
- kruskal: Kruskal-Wallace test

License

This is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

Author

- John Doe
- Jane Smith

Maintainer

- John Doe
- Jane Smith
**Details**

This particular specialization of the `plot` function is relatively simple and provided for convenience. For more advanced control of the plot it is best to use the values returned by `summary(x)`.

**Value**

None.

**References**

Please use the citation given by `citation(package="ALDEx")`.

**See Also**

`aldex, aldex.effect, aldex.ttest, aldex.glm`

**Examples**

```r
# See the examples for 'aldex'.
```

---

**aldex.set.mode**

*identify set of denominator features for log-ratio calculation*

**Description**

calculate the features that are to be used as the denominator for the Geometric Mean calculation in `clr_function.R`

**Usage**

```r
aldex.set.mode(reads, conds, denom="all")
```

**Arguments**

- **reads**: A data frame containing the samples and features per sample.
- **conds**: A vector describing which samples belong to what condition.
- **denom**: Character argument specifying which indicies to return. 'all' returns all features in both conditions. 'zero' returns the nonzero count features per condition. 'iqlr' returns the features whose variance falls within the inter-quantile range of the CLR-transformed data. In cases of malformed or null queries, input defaults to 'all'. Additionally, the input can be a numeric vector, which contains a set of row indicies to center the data against. Only for advanced users who can pre-determine the invariant set of features within their data.

**Details**

An explicit example for two conditions is shown in the ‘Examples’ below.

**Value**

Outputs a vector containing indicies per condition.
Author(s)

Jia Rong Wu

References

Please use the citation given by citation(package="ALDEx").

See Also

aldex.clr, aldex.ttest, aldex.effect, selex

Examples

# x is the output of the \code{x <- clr(data, mc.samples)} function
# conditions is a description of the data
# for the selex dataset, conditions <- c(rep("N", 7), rep("S", 7))
# input can be "all", "iqlr", "zero" or numeric for advanced users
data(selex)
#subset for efficiency
selex <- selex[1201:1600,]
cons <- c(rep("NS", 7), rep("S", 7))
x <- aldex.clr(selex, cons, mc.samples=2, denom="all")

aldex.ttest

\textit{calculate Welch's t-test and Wilcoxon test statistics}

Description

calculates expected values of the Welch's t-test and Wilcoxon rank test on the data returned by \texttt{clr\_function.r}

Usage

aldex.ttest(clr, conditions, paired.test = FALSE, hist.plot=FALSE)

Arguments

\begin{itemize}
  \item \texttt{clr} \begin{itemize}
    \item \texttt{clr} is the data output of the aldex.clr function
  \end{itemize}
  \item \texttt{conditions} \begin{itemize}
    \item a description of the data structure to be used for testing
  \end{itemize}
  \item \texttt{paired.test} \begin{itemize}
    \item whether the Welch's test should be paired or not
  \end{itemize}
  \item \texttt{hist.plot} \begin{itemize}
    \item whether to plot a histogram of P values for an individual Dirichlet Monte-Carlo instance. Plot is output to the standard R plotting device.
  \end{itemize}
\end{itemize}

Details

An explicit example for two conditions is shown in the 'Examples' below.
Value

Outputs a dataframe with the following information:

we.ep a vector containing the expected P value of the Welch’s t-test for each feature
we.eBH a vector containing the expected value of the Benjamini Hochberg corrected P value for each feature
wi.ep a vector containing the expected P value of the Wilcoxon test for each feature
wi.eBH a vector containing the expected value of the Benjamini Hochberg corrected P value for each feature

Author(s)

Greg Gloor

References

Please use the citation given by `citation(package="ALDEx")`.

See Also

`aldex.clr, aldex.glm, aldex.effect, selex`

Examples

```r
# x is the output of the `x <- aldex.clr(data, mc.samples)` function
# conditions is a description of the data
# for the selex dataset, conditions <- c(rep("N", 7), rep("S", 7))
data(selex)
# subset for efficiency
selex <- selex[1201:1600,]
conds <- c(rep("NS", 7), rep("S", 7))
x <- aldex.clr(selex, conds, mc.samples=2, denom="all")
ttest.test <- aldex.ttest(x, conds)
```
getFeatures

Details

Returns the names of the keys that can be used to subset the data rows. The keys values are the rsid’s.

Value

A vector of feature names.

See Also

aldex.clr

Examples

data(selex)
  #subset for efficiency
  selex <- selex[1201:1600,]
  conds <- c(rep("NS", 7), rep("S", 7))
  x <- aldex.clr(selex, conds, mc.samples = 2, denom="all", verbose = FALSE)
  featureNames <- getFeatureNames(x)

Description

Returns the features as a vector, for an aldex.clr object.

Usage

getFeatures(.object)

Arguments

.object A aldex.clr object containing the Monte Carlo Dirochlet instances derived from estimating the technical variance of the raw read count data, along with sample and feature information.

Details

Returns the features as a vector, for an aldex.clr object.

Value

A vector of features.

See Also

aldex.clr
getMonteCarloInstances

Examples

data(selex)
  #subset for efficiency
  selex <- selex[1201:1600,]
  conds <- c(rep("NS", 7), rep("S", 7))
  x <- aldex.clr(selex, conds, mc.samples = 2, denom = "all", verbose = FALSE)
  features <- getFeatures(x)

getMonteCarloInstances

Description

Returns the Monte Carlo Dirichlet instances used to create an aldex.clr object.

Usage

getMonteCarloInstances(.object)

Arguments

.object A aldex.clr object containing the Monte Carlo Dirichlet instances derived from estimating the technical variance of the raw read count data, along with sample and feature information.

Details

Returns the Monte Carlo Dirichlet instances used to create an aldex.clr object.

Value

A list of data frames of Monte Carlo Dirichlet instances derived from estimating the technical variance of the raw read count data.

See Also

aldex.clr

Examples

data(selex)
  #subset for efficiency
  selex <- selex[1201:1600,]
  conds <- c(rep("NS", 7), rep("S", 7))
  x <- aldex.clr(selex, conds, mc.samples = 2, denom = "all", verbose = FALSE)
  monteCarloInstances <- getMonteCarloInstances(x)
Description

Returns the designated Monte Carlo Dirochlet replicate generated from analysis, for an aldex.clr object.

Usage

getMonteCarloReplicate(.object,i)

Arguments

.object A aldex.clr object containing the Monte Carlo Dirochlet instances derived from estimating the technical variance of the raw read count data, along with sample and feature information.

i The numeric index of the desired replicate.

Details

Returns the designated Monte Carlo Dirochlet replicate generated from analysis.

Value

A data frame representing the designated Monte Carlo Dirochlet replicate generated from analysis.

See Also

aldex.clr

Examples

data(selex)
  #subset for efficiency
  selex <- selex[1201:1600,]
  conds <- c(rep("NS", 7), rep("S", 7))
  x <- aldex.clr(selex, conds, mc.samples = 2, denom = "all", verbose = FALSE)
  monteCarloInstance <- getMonteCarloReplicate(x,1)
getReads

Description

Returns the count table used as input for analysis, for an aldex.clr object.

Usage

getReads(.object)

Arguments

.object A aldex.clr object containing the Monte Carlo Dirochlet instances derived from estimating the technical variance of the raw read count data, along with sample and feature information.

Details

Returns the count table.

Value

A data frame representing the count table used as input for analysis.

See Also

aldex.clr

Examples

data(selex)
  #subset for efficiency
  selex <- selex[1201:1600,]
  conds <- c(rep("NS", 7), rep("S", 7))
  x <- aldex.clr(selex, conds, mc.samples = 2, denom = "all", verbose = FALSE)
  reads <- getReads(x)

gSampleIDs

Description

Returns the names of the samples for an aldex.clr object. These can be used to access the original reads, as in reads$sampleID (if the reads are a data frame).

Usage

gSampleIDs(.object)
numConditions

Arguments

.object A `aldex.clr` object containing the Monte Carlo Dirichlet instances derived from estimating the technical variance of the raw read count data, along with sample and feature information.

Details

Returns the number of conditions compared for analysis, for an `aldex.clr` object.

Value

A numeric representing the number of conditions compared.

details

Returns the names of the samples. These can be used to access the original reads, as in `reads$sampleID` (if the reads are a data frame).

Value

A vector of sample names.

See Also

`aldex.clr`

Examples

data(selex)
  # subset for efficiency
  selex <- selex[1201:1600,]
  conds <- c(rep("NS", 7), rep("S", 7))
  x <- aldex.clr(selex, conds, mc.samples = 2, denom = "all", verbose = FALSE)
  sampleIDs <- getSampleIDs(x)

numConditions

Description

Returns the number of conditions compared for analysis, for an `aldex.clr` object.

Usage

numConditions(.object)

Arguments

.object A `aldex.clr` object containing the Monte Carlo Dirichlet instances derived from estimating the technical variance of the raw read count data, along with sample and feature information.

Details

Returns the number of conditions compared.

Value

A numeric representing the number of conditions compared.
numFeatures

See Also
aldex.clr

Examples
data(selex)
  #subset for efficiency
  selex <- selex[1201:1600,]
  conds <- c(rep("NS", 7), rep("S", 7))
  x <- aldex.clr(selex, conds, mc.samples = 2, denom = "all", verbose = FALSE)
  conditions <- numConditions(x)

Description
Returns the number of features associated with the data, for an aldex.clr object.

Usage
numFeatures(.object)

Arguments
.object    A aldex.clr object containing the Monte Carlo Dirichlet instances derived from estimating the technical variance of the raw read count data, along with sample and feature information.

Details
Returns the number of features associated with the data.

Value
A numeric representing the number of features associated with the data.

See Also
aldex.clr

Examples
data(selex)
  #subset for efficiency
  selex <- selex[1201:1600,]
  conds <- c(rep("NS", 7), rep("S", 7))
  x <- aldex.clr(selex, conds, mc.samples = 2, denom = "all", verbose = FALSE)
  numFeatures <- numFeatures(x)
numMCInstances

Description

Returns the number of Monte Carle Dirichlet instances generated for analysis, for an aldex.clr object.

Usage

numMCInstances(.object)

Arguments

.object A aldex.clr object containing the Monte Carlo Dirichlet instances derived from estimating the technical variance of the raw read count data, along with sample and feature information.

Details

Returns the number of Monte Carle Dirichlet instances generated for analysis.

Value

A numeric representing the number of Monte Carle Dirichlet instances generated for analysis.

See Also

aldex.clr

Examples

data(selex)
  #subset for efficiency
  selex <- selex[1201:1600,]
  conds <- c(rep("NS", 7), rep("S", 7))
  x <- aldex.clr(selex, conds, mc.samples = 2, denom = "all", verbose = FALSE)
  numInstances <- numMCInstances(x)

selex

Selection-based differential sequence variant abundance dataset

Description

This data set gives the differential abundance of 1600 enzyme variants grown under selective (NS) and selective (S) conditions.

Usage

selex
Format

A dataframe of 1600 features and 14 samples. The first 7 samples are non-selected, the last 7 are selected.

Source


References

Index

*Topic classes
  - aldex.clr-class, 6
*Topic datasets
  - selex, 22
*Topic methods
  - aldex.clr-class, 6
*Topic package
  - ALDEEx2m-package, 2

aldex, 3, 13
aldex.clr, 2, 4, 9–11, 14, 15
aldex.clr, data.frame-method
  (aldex.clr), 4
aldex.clr, RangedSummarizedExperiment-method
  (aldex.clr), 4
aldex.clr-class, 6
aldex.clr.function, 7
aldex.clr.function (aldex.clr), 4
aldex.corr, 4, 8
aldex.effect, 2, 4, 5, 9, 11, 13–15
aldex glm, 2, 4, 5, 9, 10, 10, 13, 15
aldex.plot, 12
aldex.set.mode, 13
aldex.ttest, 2, 4, 5, 10, 11, 13, 14, 15
ALDEEx2m (ALDEEx2m-package), 2
ALDEEx2m-package, 2

getFeatureNames, 15
getFeatureNames, aldex.clr-method
  (getFeatureNames), 15
getFeatures, 16
getFeatures, aldex.clr-method
  (getFeatures), 16
getMonteCarloInstances, 17
getMonteCarloInstances, aldex.clr-method
  (getMonteCarloInstances), 17
getMonteCarloReplicate, 18
getMonteCarloReplicate, aldex.clr, numeric-method
  (getMonteCarloReplicate), 18
getReads, 19
getReads, aldex.clr-method (getReads), 19
getSampleIDs, 19
getSampleIDs, aldex.clr-method
  (getSampleIDs), 19

numConditions, 20
numConditions, aldex.clr-method
  (numConditions), 20
numFeatures, 21
numFeatures, aldex.clr-method
  (numFeatures), 21
numMCInstances, 22
numMCInstances, aldex.clr-method
  (numMCInstances), 22

selex, 2, 4, 5, 9–11, 14, 15, 22