Package ‘MetaLonDA’

October 12, 2022

Type Package

Title Metagenomics Longitudinal Differential Abundance Method

Version 1.1.8

Date 2019-12-17

Author Ahmed A. Metwally, Yang Dai, Patricia W. Finn, David L. Perkins

Maintainer Ahmed A. Metwally <`ametwall@stanford.edu`>

URL `https://github.com/aametwally/MetaLonDA`

BugReports `https://github.com/aametwally/MetaLonDA/issues`

Description Identify time intervals of differentially abundant metagenomic features in longitudinal studies (Metwally AA, et al., Microbiome, 2018 <doi:10.1186/s40168-018-0402-y>).

License MIT + file LICENSE

Depends R(>= 3.5.0)

Imports gss, plyr, zoo, pracma, ggplot2, parallel, doParallel, metagenomeSeq, DESeq2, edgeR

Suggests knitr, rmarkdown

biocViews

Repository CRAN

NeedsCompilation no

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

VignetteBuilder knitr

Date/Publication 2019-12-18 02:10:26 UTC

R topics documented:

`areaPermutation` *Calculate Area Ratio (AR) of each feature's time interval for all permutations*

Description

Fits longitudinal samples from the same group using negative binomial or LOWESS for all permutations

Usage

```
areaPermutation(perm)
```

Arguments

`perm`	list has all the permutated models

Value

returns a list of all permutation area ratio

References

Ahmed Metwally (ametwall@stanford.edu)

Examples

```
data(metalonda_test_data)
n.sample = 5 # sample size;
n.timepoints = 10 # time point;
n.perm = 3
n.group= 2 # number of group;
Group = factor(c(rep(0,n.sample*n.timepoints), rep(1,n.sample*n.timepoints)))
Time = rep(rep(1:n.timepoints, times = n.sample), 2)
ID = factor(rep(1:(2*n.sample), each = n.timepoints))
points = seq(1, 10, length.out = 10)
aggretage.df = data.frame(Count = metalonda_test_data[1,], Time = Time, Group = Group, ID = ID)
perm = permutation(aggretage.df, n.perm = 3, method = "nbinomial", points)
areaPermutation(perm)
```

`curveFitting` *Fit longitudinal data*

Description

Fits longitudinal samples from the same group using negative binomial smoothing splines or LOWESS

Usage

```
curveFitting(df, method = "nbinomial", points)
```

Arguments

`df`	dataframe has the Count, Group, ID, Time
`method`	fitting method (nbinomial, lowess)
`points`	points at which the prediction should happen

Value

returns the fitted model

References

Ahmed Metwally (ametwall@stanford.edu)

Examples

```
data(metalonda_test_data)
n.sample = 5
n.timepoints = 10
n.group = 2
Group = factor(c(rep(0, n.sample*n.timepoints), rep(1, n.sample*n.timepoints)))
Time = rep(rep(1:n.timepoints, times = n.sample), 2)
ID = factor(rep(1:(2*n.sample), each = n.timepoints))
```

```
points = seq(1, 10, length.out = 10)
aggretage.df = data.frame(Count = metalonda_test_data[1,], Time = Time, Group = Group, ID = ID)
cf = curveFitting(df = aggretage.df, method= "nbinomial", points)
```

`findSigInterval` *Find significant time intervals*

Description

Identify significant time intervals

Usage

```
findSigInterval(adjusted.pvalue, threshold = 0.05, sign)
```

Arguments

`adjusted.pvalue`	vector of the adjusted p-value
`threshold`	p-value cut off
`sign`	vector hold area sign of each time interval

Value

returns a list of the start and end points of all significant time intervals

References

Ahmed Metwally (ametwall@stanford.edu)

Examples

```
p = c(0.04, 0.01, 0.02, 0.04, 0.06, 0.2, 0.06, 0.04)
sign = c(1, 1, 1, 1, -1, -1, 1, 1)
findSigInterval(p, threshold = 0.05, sign)
```

---

intervalArea *Calculate Area Ratio (AR) of each feature's time interval*

Description

Calculate Area Ratio (AR) of each feature's time interval

Usage

```
intervalArea(curve.fit.df)
```

Arguments

`curve.fit.df`	gss data object of the fitted spline

Value

returns the area ratio for all time intervals

References

Ahmed Metwally (ametwall@stanford.edu)

Examples

```
data(metalonda_test_data)
n.sample = 5
n.timepoints = 10
n.group= 2
Group = factor(c(rep(0,n.sample*n.timepoints), rep(1,n.sample*n.timepoints)))
Time = rep(rep(1:n.timepoints, times = n.sample), 2)
ID = factor(rep(1:(2*n.sample), each = n.timepoints))
points = seq(1, 10, length.out = 10)
aggregate.df = data.frame(Count = metalonda_test_data[1,], Time = Time, Group = Group, ID = ID)
model = curveFitting(df = aggregate.df, method= "nbinomial", points)
intervalArea(model)
```

---

metalonda *Metagenomic Longitudinal Differential Abundance Analysis for one feature*

Description

Find significant time intervals of the one feature

Usage

```
metalonda(
  Count,
  Time,
  Group,
  ID,
  n.perm = 500,
  fit.method = "nbinomial",
  points,
  text = 0,
  parall = FALSE,
  pvalue.threshold = 0.05,
  adjust.method = "BH",
  time.unit = "days",
  ylabel = "Normalized Count",
  col = c("blue", "firebrick"),
  prefix = "Test"
)
```

Arguments

`Count`	matrix has the number of reads that mapped to each feature in each sample.
`Time`	vector of the time label of each sample.
`Group`	vector of the group label of each sample.
`ID`	vector of the subject ID label of each sample.
`n.perm`	number of permutations.
`fit.method`	fitting method (nbinomial, lowess).
`points`	points at which the prediction should happen.
`text`	Feature's name.
`parall`	boolean to indicate whether to use multicore.
`pvalue.threshold`	p-value threshold cutoff for identifing significant time intervals.
`adjust.method`	multiple testing correction method.
`time.unit`	time unit used in the Time vector (hours, days, weeks, months, etc.)
`ylabel`	text to be shown on the y-axis of all generated figures (default: "Normalized Count")
`col`	two color to be used for the two groups (eg., c("red", "blue")).
`prefix`	prefix to be used to create directory for the analysis results

Value

returns a list of the significant time intervals for the tested feature.

References

Ahmed Metwally (ametwall@stanford.edu)

Examples

```
data(metalonda_test_data)
n.sample = 5
n.timepoints = 10
n.group = 2
Group = factor(c(rep(0, n.sample*n.timepoints), rep(1,n.sample*n.timepoints)))
Time = rep(rep(1:n.timepoints, times = n.sample), 2)
ID = factor(rep(1:(2*n.sample), each = n.timepoints))
points = seq(1, 10, length.out = 10)
## Not run:
output.nbinomial = metalonda(Count = metalonda_test_data[1,], Time = Time, Group = Group,
ID = ID, fit.method =  "nbinomial", n.perm = 10, points = points,
text = rownames(metalonda_test_data)[1], parall = FALSE, pvalue.threshold = 0.05,
adjust.method = "BH", time.unit = "hours", ylabel = "Normalized Count", col = c("black", "green"))

## End(Not run)
```

`metalondaAll` *Metagenomic Longitudinal Differential Abundance Analysis for all Features*

Description

Identify significant features and their significant time interval

Usage

```
metalondaAll(
  Count,
  Time,
  Group,
  ID,
  n.perm = 500,
  fit.method = "nbinomial",
  num.intervals = 100,
  parall = FALSE,
  pvalue.threshold = 0.05,
  adjust.method = "BH",
  time.unit = "days",
  norm.method = "none",
  prefix = "Output",
  ylabel = "Normalized Count",
  col = c("blue", "firebrick")
)
```

Arguments

`Count`	Count matrix of all features
`Time`	Time label of all samples
`Group`	Group label of all samples
`ID`	individual ID label for samples
`n.perm`	number of permutations
`fit.method`	The fitting method (nbinomial, lowess)
`num.intervals`	The number of time intervals at which metalonda test differential abundance
`parall`	logic to indicate whether to use multicore
`pvalue.threshold`	p-value threshold cutoff
`adjust.method`	Multiple testing correction methods
`time.unit`	time unit used in the Time vector (hours, days, weeks, months, etc.)
`norm.method`	normalization method to be used to normalize count matrix (css, tmm, ra, log10, median_ratio)
`prefix`	prefix for the output figure
`ylabel`	text to be shown on the y-axis of all generated figures (default: "Normalized Count")
`col`	two color to be used for the two groups (eg., c("red", "blue")).

Value

Returns a list of the significant features a long with their significant time intervals

References

Ahmed Metwally (ametwall@stanford.edu)

Examples

```
## Not run:
data(metalonda_test_data)
n.sample = 5
n.timepoints = 10
n.group = 2
Group = factor(c(rep(0, n.sample*n.timepoints), rep(1,n.sample*n.timepoints)))
Time = rep(rep(1:n.timepoints, times = n.sample), 2)
ID = factor(rep(1:(2*n.sample), each = n.timepoints))
points = seq(1, 10, length.out = 10)
output.nbinomial = metalondaAll(Count = metalonda_test_data, Time = Time, Group = Group,
ID = ID, n.perm = 10, fit.method =  ”nbinomial”, num.intervals = 100,
parall = FALSE, pvalue.threshold = 0.05, adjust.method = ”BH”,
time.unit = ”hours”, norm.method = ”none”, prefix = ”Test”,  time.unit = ”hours”,
ylabel = ”Normalized Count”, col = c(”black”, ”green”))

## End(Not run)
```

`metalonda_test_data` *Simulated data of OTU abundance for 2 phenotypes each has 5 subjects at 10 time-points*

Description

The dataset is used for testing the MetaLonDA

Usage

```
metalonda_test_data
```

Format

A data frame with 2 OTUs patterns

`normalize` *Normalize count matrix*

Description

Normalize count matrix

Usage

```
normalize(count, method = "css")
```

Arguments

`count`	count matrix
`method`	normalization method

References

Ahmed Metwally (ametwall@stanford.edu)

`permutation` *Permute group labels*

Description

Permutes the group label of the samples in order to construct the AR empirical distibution

Usage

```
permutation(
  perm.dat,
  n.perm = 500,
  method = "nbinomial",
  points,
  lev,
  parall = FALSE
)
```

Arguments

`perm.dat`	dataframe has the Count, Group, ID, Time
`n.perm`	number of permutations
`method`	The fitting method (negative binomial, LOWESS)
`points`	The points at which the prediction should happen
`lev`	the two level's name
`parall`	boolean to indicate whether to use multicore.

Value

returns the fitted model for all the permutations

References

Ahmed Metwally (ametwall@stanford.edu)

Examples

```
data(metalonda_test_data)
n.sample = 5
n.timepoints = 10
n.perm = 3
n.group = 2
Group = factor(c(rep(0, n.sample*n.timepoints), rep(1, n.sample*n.timepoints)))
Time = rep(rep(1:n.timepoints, times = n.sample), 2)
ID = factor(rep(1:(2*n.sample), each = n.timepoints))
points = seq(1, 10, length.out = 10)
aggregate.df = data.frame(Count = metalonda_test_data[1,], Time = Time, Group = Group, ID = ID)
prm = permutation(aggregate.df, n.perm = 3, method = "nbinomial", points)
```

`visualizeArea` *Visualize significant time interval*

Description

Visualize significant time interval

Usage

```
visualizeArea(
  aggregate.df,
  model.ss,
  method,
  start,
  end,
  text,
  group.levels,
  unit = "days",
  ylabel = "Normalized Count",
  col = c("blue", "firebrick"),
  prefix = "Test"
)
```

Arguments

`aggregate.df`	Dataframe has the Count, Group, ID, Time
`model.ss`	The fitted model
`method`	Fitting method (negative binomial or LOWESS)
`start`	Vector of the start points of the time intervals
`end`	Vector of the end points of the time intervals
`text`	Feature name
`group.levels`	Level's name
`unit`	time unit used in the Time vector (hours, days, weeks, months, etc.)
`ylabel`	text to be shown on the y-axis of all generated figures (default: "Normalized Count")
`col`	two color to be used for the two groups (eg., c("red", "blue")).
`prefix`	prefix to be used to create directory for the analysis results

References

Ahmed Metwally (ametwall@stanford.edu)

`visualizeARHistogram` *Visualize Area Ratio (AR) empirical distribution*

Description

Visualize Area Ratio (AR) empirical distribution for each time interval

Usage

```
visualizeARHistogram(permuted, text, method, prefix = "Test")
```

Arguments

`permuted`	Permutation of the permuted data
`text`	Feature name
`method`	fitting method
`prefix`	prefix to be used to create directory for the analysis results

References

Ahmed Metwally (ametwall@stanford.edu)

`visualizeFeature` *Visualize Longitudinal Feature*

Description

Visualize Longitudinal Feature

Usage

```
visualizeFeature(
  df,
  text,
  group.levels,
  unit = "days",
  ylabel = "Normalized Count",
  col = c("blue", "firebrick"),
  prefix = "Test"
)
```

Arguments

`df`	dataframe has the Count, Group, ID, Time
`text`	feature name
`group.levels`	The two level's name
`unit`	time interval unit
`ylabel`	text to be shown on the y-axis of all generated figures (default: "Normalized Count")
`col`	two color to be used for the two groups (eg., c("red", "blue")).
`prefix`	prefix to be used to create directory for the analysis results

References

Ahmed Metwally (ametwall@stanford.edu)

Examples

```
data(metalonda_test_data)
pfx = tempfile()
dir.create(file.path(pfx))
n.sample = 5
n.timepoints = 10
n.group = 2
Group = factor(c(rep(0, n.sample*n.timepoints), rep(1, n.sample*n.timepoints)))
Time = rep(rep(1:n.timepoints, times = n.sample), 2)
ID = factor(rep(1:(2*n.sample), each = n.timepoints))
points = seq(1, 10, length.out = 10)
aggregate.df = data.frame(Count = metalonda_test_data[1,], Time = Time, Group = Group, ID = ID)
visualizeFeature(df = aggregate.df, text = rownames(metalonda_test_data)[1],
group.levels = Group, prefix = pfx)
```

`visualizeFeatureSpline` *Visualize the feature trajectory with the fitted Splines*

Description

Plot the longitudinal features along with the fitted splines

Usage

```
visualizeFeatureSpline(
  df,
  model,
  method,
  text,
  group.levels,
```

```
  unit = "days",
  ylabel = "Normalized Count",
  col = c("blue", "firebrick"),
  prefix = "Test"
)
```

Arguments

`df`	dataframe has the Count , Group, ID, Time
`model`	the fitted model
`method`	The fitting method (negative binomial, LOWESS)
`text`	feature name
`group.levels`	The two level's name
`unit`	time unit used in the Time vector (hours, days, weeks, months, etc.)
`ylabel`	text to be shown on the y-axis of all generated figures (default: "Normalized Count")
`col`	two color to be used for the two groups (eg., c("red", "blue")).
`prefix`	prefix to be used to create directory for the analysis results

References

Ahmed Metwally (ametwall@stanford.edu)

`visualizeTimeIntervals`

Visualize all significant time intervals for all tested features

Description

Visualize all significant time intervals for all tested features

Usage

```
visualizeTimeIntervals(
  interval.details,
  prefix = "Test",
  unit = "days",
  col = c("blue", "firebrick")
)
```

Arguments

`interval.details`	Dataframe has infomation about significant interval (feature name, start, end, dominant, p-value)
`prefix`	prefix for the output figure
`unit`	time unit used in the Time vector (hours, days, weeks, months, etc.)
`col`	two color to be used for the two groups (eg., c("red", "blue")).

References

Ahmed Metwally (ametwall@stanford.edu)

`visualizeVolcanoPlot` *Visualize log2 fold-change and significance of each interval as volcano plot*

Description

Visualize log2 fold-change and significance of each interval as volcano plot

Usage

```
visualizeVolcanoPlot(df, text, prefix = "Test")
```

Arguments

`df`	Dataframe has a detailed summary about feature’s significant intervals
`text`	Feature name
`prefix`	prefix to be used to create directory for the analysis results

References

Ahmed Metwally (ametwall@stanford.edu)

Index