Package: Rnits (via r-universe)

November 1, 2026

Type Package

Title R Normalization and Inference of Time Series data

Version 1.47.0

Date 2022-11-04

Author Dipen P. Sangurdekar <dipen.sangurdekar@gmail.com>

Maintainer Dipen P. Sangurdekar <dipen.sangurdekar@gmail.com>

Depends R (>= 3.6.0), Biobase, ggplot2, limma, methods

Imports affy, boot, impute, splines, graphics, qvalue, reshape2

Suggests BiocStyle, knitr, GEOquery, stringr

Description R/Bioconductor package for normalization, curve registration and inference in time course gene expression data.

biocViews GeneExpression, Microarray, TimeCourse, DifferentialExpression, Normalization

Lazyload yes

LazyData yes

License GPL-3

VignetteBuilder knitr

NeedsCompilation no

Config/pak/sysreqs libicu-dev zlib1g-dev

Repository https://bioc.r-universe.dev

Date/Publication 2026-04-28 12:39:39 UTC

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

RemoteRef HEAD

RemoteSha e2d32d68424ff070882d213188de764994e1b717

Contents

$ . . . 2
build.Rnits . . . 2
calculateGCV . . . 4
fit . . . 5
getCID . . . 6
getFitModel . . . 7
getLR . . . 8
getNormTwoChannel . . . 8
getPval . . . 9
getStat . . . 10
plotResults . . . 10
rnits . . . 11
Rnits-class . . . 12
summarizeProbes . . . 12
summary,Rnits-method . . . 13
timeAlign . . . 14
topData . . . 15
yeastchemostat . . . 16

Index **17**

`$` *replace slot of Rnits*

Description

replace slot of Rnits

`build.Rnits` *Input the RGlist raw data, build a Rnits object and perform filtering and normalization*

Description

This function takes high-dimensional expression data as a RGList, creates a `Rnits` object for subsequent filtering and normalization

Usage

```
build.Rnits(obj, probedata = NULL, phenodata = NULL, filter = NULL,
  normalize = NULL, normmethod = NULL, plot = FALSE, center = FALSE,
  background = NULL, threshold = 0.8, logscale = FALSE)
```

Arguments

obj	Raw expression data in RGlist, AffyBatch or simple data frame format
probedata	A data frame containing the probe names that should match the probe names in raw data (optional)
phenodata	A data frame with information about sample names. The rownames of the data frame must match column names of the expression values. If input data is data frame of log ratios, this is required.
filter	An argument to perform background filtering of probes. If NULL, no filtering is done. If an integer (0-500), probes are flagged based on raw channel intensity. If a vector of two numbers is provided, the first will be used for red channel and the second for green channel. If 'background', probes whose intensities are lower than 2 standard deviations less than the mean of the background intensity for the channel are flagged.
normalize	Character string specifying the normalization method for raw data. If Intensity, the reference channels for all arrays are used to construct an array-specific smoothing function which is then applied to normalize the sample channel. If Between, the normalization method normalizeBetweenArrays in the LIMMA package is used (use normmethod to further specify normalization methods. See packaged LIMMA for details.). If Within, the normalization method normalizeWithinArrays in the LIMMA package is used.
normmethod	Normalization method for input data. Default NULL. Can be one of 'quantile', 'vsn', 'Between'
background	Only for AffyBatch data. If TRUE, background filtering will be done on Affy data.
center	If TRUE, the log-ratio data will be mean centered to 0 in the column space.
plot	If TRUE, boxplots of normalized channel intensities and log-ratios are drawn.
threshold	Default 0.8. Fraction of samples with missing data for individual probes to be filtered out.
logscale	Default FALSE. Is the data in logscale? If FALSE, log2 transformation is done on the data.

Details

See the Limma User's Guide for more details on read.maimages, normalizeBetweenArrays, normalizeWithinArrays and RGList. For importing microarray raw data, use the 'Targets file' to specify experimental design. The target file has columns SlideNumber, FileName, Cy3 (description of Cy3 channel ref/control/treatment), Cy5 (description of Cy3 channel ref/control/treatment) and Time. Time values should be identical for control and treatment.

Value

An object of S4 class Rnits (which is derived from class exprSet), containing the probe data, design data, expression data, phenotypical data (i.e. Time).

See Also

ExpressionSet

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
```

`calculateGCV` *Calculate the optimal B-spline model using generalized cross-validation*

Description

Calculate the optimal B-spline model using generalized cross-validation

Usage

```
calculateGCV(object, topcomp = 5)

## S4 method for signature 'Rnits'
calculateGCV(object, topcomp = 5)
```

Arguments

`object`	`Rnits` object
`topcomp`	The number of top eigenvectors to be used for computation

Details

The optimal B-spline model is chosen as the largest model that minimizes the cross validation error of the top N eigenvectors of each time series data.

Value

A list object with fields 'degree', 'df' for each time series data set.

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
opt_model <- calculateGCV(rnitsobj)
## Not run:
rnitsobj <- fit(rnitsobj, gene.level = TRUE, model = opt.model)

## End(Not run)
```

`fit` *Fit model on time series data*

Description

Fit a model comparing time series data set `Rnits` objects

Usage

```
fit(object, cluster = TRUE, B = 100, verbatim = FALSE, nclus = NULL,
  modelhistplot = FALSE, seed = 123, gene.level = TRUE,
  clusterallsamples = FALSE, model = NULL)

## S4 method for signature 'Rnits'
fit(object, cluster = TRUE, B = 100, verbatim = FALSE,
  nclus = NULL, modelhistplot = FALSE, seed = 123, gene.level = TRUE,
  clusterallsamples = FALSE, model = NULL)
```

Arguments

`object`	`Rnits` object
`cluster`	if `TRUE`, perform clustering to identify groups of genes/probes with similar expression profiles.
`B`	Default `100`. Number of bootstrap iterations for p-value calculation
`verbatim`	If `FALSE`, print out details of fitting models.
`nclus`	Default `NULL`. Number of clusters to use for k-means clustering.
`modelhistplot`	If `TRUE`, p-value histograms of multiple models are plotted.
`seed`	Random seed for bootstrap iterations
`gene.level`	If `TRUE`, collapse probes to gene level information.
`clusterallsamples`	If `TRUE`, Use all time series for clustering. By default, only the sample labeled 'control' is used or the lexically first sample is used.
`model`	A data frame with fields 'degree' and 'df' indicating a specific B-spline model to be used. If provided, model selection is not run.

Details

The function compares multiple time-series expression data sets by i) (optional) summarizing probes into gene-level information ii) (optional) identifying a set of co-expressed genes by clustering iii) For each cluster (or for all genes /probes), fit a series of B-splines with varying curvature and degrees of freedom. Under the null hypothesis `H_0`, a single model is fit for all data sets, while under `H_1`, each data set is fit separately. P-values from the hypothesis test are then plotted and the least complex spline parameters that result in uniformly distributed null p-values are automatically chosen.

Value

An object of S4 class Rnits with fitted results data containing cluster information, ratio statistics and p-values.

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
## Not run:
# Fit model using gene-level summarization
rnitsobj <- fit(rnitsobj, gene.level = TRUE, clusterallsamples = FALSE)

## End(Not run)
```

---

getCID *Cluster IDs of probes/genes from fitted Rnits*

Description

Retrieve cluster IDs of probes/genes from fitted Rnits object after fit has been run.

Usage

```
getCID(object)

## S4 method for signature 'Rnits'
getCID(object)
```

Arguments

object	Rnits

Details

If cluster = False during fitting, a vector of 1s will be returned.

Value

A vector of cluster IDs corresponding to gene/probe names

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
## Not run:
# Fit model using gene-level summarization
rnitsobj <- fit(rnitsobj, gene.level = TRUE, clusterallsamples = FALSE)
```

```
# Get cluster IDs from fitted model
cid <- getCID(rnitsobj)

## End(Not run)
```

---

getFitModel *Extract fit data from* Rnits *object*

Description

Retrieve model fit data from Rnits object after fit has been run.

Usage

```
getFitModel(object)

## S4 method for signature 'Rnits'
getFitModel(object)
```

Arguments

object	Rnits

Details

Contains Ratio statistic, p-value and cluster ID data

Value

A data frame containing the model fit results for all genes

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
## Not run:
# Fit model using gene-level summarization
rnitsobj <- fit(rnitsobj, gene.level = TRUE, clusterallsamples = FALSE)

# P-values, ratio statistics and cluster ID's can be retrieved for all genes together
fitdata <- getFitModel(rnitsobj)

## End(Not run)
```

getLR *Get log-ratios*

Description

Extract normalized log-ratios from Rnits object

Usage

```
getLR(object, impute = FALSE)

## S4 method for signature 'Rnits'
getLR(object, impute = FALSE)
```

Arguments

object	Rnits object
impute	If TRUE, perform K-NN imputation to fill missing values

Value

A matrix of normalized log-ratios.

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
## Not run:
# Fit model using gene-level summarization
rnitsobj <- fit(rnitsobj, gene.level = TRUE, clusterallsamples = FALSE)

# Get logratios
lr <- getLR(rnitsobj)

## End(Not run)
```

getNormTwoChannel *Get Normalized channel data for two channel arrays*

Description

For two color data, extract normalized channel data from Rnits object

Usage

```
getNormTwoChannel(object)

## S4 method for signature 'Rnits'
getNormTwoChannel(object)
```

Arguments

object	Rnits object

Value

A list containing R and G fields for normalized Red and Green channel data respectively.

---

getPval *Get p-values*

Description

Extract p-values from fitted Rnits object

Usage

```
getPval(object)

## S4 method for signature 'Rnits'
getPval(object)
```

Arguments

object	Rnits object

Value

An vector of p-values

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
## Not run:
# Fit model using gene-level summarization
rnitsobj <- fit(rnitsobj, gene.level = TRUE, clusterallsamples = FALSE)

#Get pvalues from fitted model
pval <- getPval(rnitsobj)


## End(Not run)
```

getStat *Retrieve ratio statistics*

Description

Extract ratio statistics from fitted Rnits object

Usage

```
getStat(object)

## S4 method for signature 'Rnits'
getStat(object)
```

Arguments

object	Rnits object

Value

An vector of ratio statistics

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
## Not run:
# Fit model using gene-level summarization
rnitsobj <- fit(rnitsobj, gene.level = TRUE, clusterallsamples = FALSE)

# Get ratio statistics from fitted model
stat <- getStat(rnitsobj)

## End(Not run)
```

plotResults *Plot profiles of top genes/probes*

Description

After fit has been applied on Rnits object, plot the profiles of N top ranking genes/probes.

Usage

```
plotResults(object, id = NULL, fdr = NULL, top = 48, pdf = FALSE,
  sort.by = "p-value", filename = "TopPlots.pdf", scale_y = NULL)

## S4 method for signature 'Rnits'
plotResults(object, id = NULL, fdr = NULL, top = 48,
  pdf = FALSE, sort.by = "p-value", filename = "TopPlots.pdf",
  scale_y = NULL)
```

Arguments

object	Rnits object.
id	Names of specific genes or probes to be plotted. Overrides fdr and top argument.
fdr	FDR cut-off plotting top probes or genes. Overrides top argument.
top	Number of top genes or probes whose profile is to be plotted. Default 48.
pdf	Save plot as pdf? Default FALSE.
sort.by	Criteria for sorting top genes or probes. Default 'p-value'.
filename	Name of pdf file. Default topplots.pdf.
scale_y	If 'free', use free scales for plots. Default NULL.
...	Optional arguments to plot

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
## Not run:
# Fit model using gene-level summarization
rnitsobj <- fit(rnitsobj, gene.level = TRUE, clusterallsamples = FALSE)

# Plot top results
plotResults(rnitsobj, top = 16)


## End(Not run)
```

---

rnits *rnits.*

Description

rnits.

Rnits-class *rnits class*

Description

Class rnits for time series

Details

Some details

summarizeProbes *Summarize probe level data to gene level data*

Description

The code utilizes the probe-gene mapping from features file to summarize probe-level log ratios to gene level ratios.

Usage

```
summarizeProbes(object)

## S4 method for signature 'Rnits'
summarizeProbes(object)
```

Arguments

object	Rnits object

Details

Tukey's biweight is used to compute gene level summary

Value

An object of class Rnits with gene level log ratios, which can be retrieved by `getLR(object)`

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')

# Summarize gene-level data
rnitsobj <- summarizeProbes(rnitsobj)
```

summary,Rnits-method *Summary of fit*

Description

Summarize top genes or probes from Rnits `fit` method

Usage

```
## S4 method for signature 'Rnits'
summary(object, top = 48, fdr = NULL, plot = FALSE,
  sort.by = "p-value")
```

Arguments

`object`	`Rnits` object on which `fit` has been applied
`top`	Display results for top N genes/probes. Default `50`
`fdr`	Display results for genes/probes less than FDR (%) cutoff (if provided). Overrides `top` argument
`plot`	If `TRUE`, plot histogram of p-values
`sort.by`	Sort top results by either `p-value` or `FDR`

Value

A table of top genes/profiles listing the ratio statistics, p-values, q-values and cluster information.

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
## Not run:
# Fit model using gene-level summarization
rnitsobj <- fit(rnitsobj, gene.level = TRUE, clusterallsamples = FALSE)

# Get summary of top genes
summary(rnitsobj, FDR = 5)


## End(Not run)
```

timeAlign *Curve registration of time series curves*

Description

Align multiple time series to the average seris

Usage

```
timeAlign(object, iterMax = 5, seed = 123, null.frac = 0.75,
  anchor = NULL, rerun = FALSE, plot = FALSE)

## S4 method for signature 'Rnits'
timeAlign(object, iterMax = 5, seed = 123,
  null.frac = 0.75, anchor = NULL, rerun = FALSE, plot = FALSE)
```

Arguments

object	rnits object
iterMax	Maximum iterations to be performed
seed	Random seed
null.frac	Fraction of genes that are considered as null
anchor	Sample to be considerded as base for aligning time series. If not provided, the average is used
rerun	If TRUE, re-align previously aligned data
plot	If TRUE, plot results

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')

# Do curve-registration on data
rnitsobj <- timeAlign(rnitsobj)
```

topData *Data of top genes/probes*

Description

Extract expression data for top genes/probes

Usage

```
topData(object, id = NULL, fdr = NULL, top = 16, sort.by = "p-value")

## S4 method for signature 'Rnits'
topData(object, id = NULL, fdr = NULL, top = 16,
  sort.by = "p-value")
```

Arguments

object	Rnits object on which fit has been applied
id	Names of probes or genes
top	Display results for top N genes/probes. Default 50
fdr	Display results for genes/probes less than FDR cutoff (if provided). Overrides top argument
sort.by	Sort top results by either p-value or FDR

Value

A table of expression values of top genes/profiles

Examples

```
# load pre-compiled expressionSet object for Ronen and Botstein yeast  chemostat data
data(yeastchemostat)
rnitsobj = build.Rnits(yeastchemostat, logscale = TRUE, normmethod = 'Between')
## Not run:
# Fit model using gene-level summarization
rnitsobj <- fit(rnitsobj, gene.level = TRUE, clusterallsamples = FALSE)

#Get data for top genes
td <- topData(rnitsobj, FDR = 5)


## End(Not run)
```

`yeastchemostat` *Yeast chemostat data from Ronen and Botstein (Proc Natl Acad Sci U S A. 2006 Jan 10;103(2):389-94. Epub 2005 Dec 28.)*

Description

(From author's GEO submission) Transcriptional response of steady-state yeast cultures to transient perturbations in carbon source

Usage

```
yeastchemostat
```

Format

An `ExpressionSet` object with containing 'Sample' and 'Time' columns and replicates removed.

Source

http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE4158

Index

$, 2
$<-,Rnits-method *($)*, 2

build.Rnits, 2

calculateGCV, 4
calculateGCV,character,ANY-method *(calculateGCV)*, 4
calculateGCV,Rnits-method *(calculateGCV)*, 4

exprSet, *3*

fit, 5
fit,character,ANY-method *(fit)*, 5
fit,Rnits-method *(fit)*, 5

getCID, 6
getCID,character,ANY-method *(getCID)*, 6
getCID,Rnits-method *(getCID)*, 6
getFitModel, 7
getFitModel,character,ANY-method *(getFitModel)*, 7
getFitModel,Rnits-method *(getFitModel)*, 7
getLR, 8
getLR,character,ANY-method *(getLR)*, 8
getLR,Rnits-method *(getLR)*, 8
getNormTwoChannel, 8
getNormTwoChannel,character,ANY-method *(getNormTwoChannel)*, 8
getNormTwoChannel,Rnits-method *(getNormTwoChannel)*, 8
getPval, 9
getPval,character,ANY-method *(getPval)*, 9
getPval,Rnits-method *(getPval)*, 9
getStat, 10
getStat,character,ANY-method *(getStat)*, 10
getStat,Rnits-method *(getStat)*, 10

plotResults, 10
plotResults,character,ANY-method *(plotResults)*, 10
plotResults,Rnits-method *(plotResults)*, 10

Rnits, *2–13*, *15*
rnits, 11
Rnits-class, 12
rnits-package *(rnits)*, 11
Rnits.getLR *(getLR)*, 8

summarizeProbes, 12
summarizeProbes,character,ANY-method *(summarizeProbes)*, 12
summarizeProbes,Rnits-method *(summarizeProbes)*, 12
summary,Rnits-method, 13

timeAlign, 14
timeAlign,character,ANY-method *(timeAlign)*, 14
timeAlign,Rnits-method *(timeAlign)*, 14
topData, 15
topData,character,ANY-method *(topData)*, 15
topData,Rnits-method *(topData)*, 15

yeastchemostat, 16