Package 'Trendy'

March 11, 2025

Type Package

Title Breakpoint analysis of time-course expression data

Version 1.29.0

Author Rhonda Bacher and Ning Leng

Maintainer Rhonda Bacher <rbacher@ufl.edu>

Description Trendy implements segmented (or breakpoint) regression models to estimate breakpoints which represent changes in expression for each feature/gene in high throughput data with ordered conditions.

Depends R (>= 3.4)

Imports stats, utils, graphics, grDevices, segmented, gplots, parallel, magrittr, BiocParallel, DT, S4Vectors, SummarizedExperiment, methods, shiny, shinyFiles

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Suggests BiocStyle, knitr, rmarkdown, devtools

VignetteBuilder knitr

biocViews TimeCourse, RNASeq, Regression, ImmunoOncology

URL https://github.com/rhondabacher/Trendy

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

git_branch devel

git_last_commit e2f06d3

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-03-10

Contents

`breakpointDist` *Distribution of breakpoints*

Description

calculates number of breakpoints at each time.

Usage

```
breakpointDist(topTrendyData, NDigits = 0)
```

Arguments

`topTrendyData`	results from topTrendy() function
`NDigits`	how many digits to be used when rounding (default is 0 (return integers))

Value

The function takes significant genes called from the topTrendyData() function. For any time point, this function calculates how many genes have a breakpoint at this time point. The output is the numbers of genes sorted by time point.

Author(s)

Ning Leng

Examples

```
m1 <- matrix(c(c(rnorm(50,5,1),sort(rnorm(50, 15, 5))), rnorm(100, 50,10)), 2, 100, TRUE)
rownames(m1) <- c("g1","g2")
colnames(m1) <- paste0("time", seq_len(100))
myTrends <- results(trendy(m1))
topGenes <- topTrendy(myTrends)
bpDist <- breakpointDist(topGenes)
```

`breakpointFit` *break point fits*

Description

break point fits

Usage

```
breakpointFit(J, tVectIn, lmLinear, numTry)
```

Arguments

`J`	number of breakpoints in the model
`tVectIn`	a numerical vector indicating the time-points or the order of samples. If it is NULL (default), then the time/order will be assumed to be equaly spaced from 1:N (N is number of samples).
`lmLinear`	the linear model fit; no breakpoints
`numTry`	the number of different seeds to try. If all numTry runs fail, then the linear regression (no breakpoints, one segment) model will be returned.

`extractPattern` *Extract pattern from segmented regression*

Description

find dynamic genes that follow a given pattern

Usage

```
extractPattern(trendyOutData, Pattern = NULL, adjR2Cut = 0.5,
  Delay = 0)
```

Arguments

trendyOutData	output from trendy() function
Pattern	vector containing pattern to search genes/features (e.g, c("up", "down")), no-change is designated by "same". If length is one (e.g c("up")) then it will only consider features with constant pattern across the entire time-course.
adjR2Cut	only consider features with adjusted R^2 > adjR2Cut. Default = .5.
Delay	search for pattern starting after certain time-point (e.g. only genes with a break-point > 10).

Value

Genes: names of genes/features containing pattern and the breakpoints corresponding to the pattern.

Author(s)

Rhonda Bacher

Examples

```
myTrends <- trendy(trendyExampleData[seq_len(5),], tVect=seq_len(40))
myTrends <- results(myTrends)
#extractPattern(myTrends, Pattern = c("up")) #increasing only features
#extractPattern(myTrends, Pattern = c("same", "down"))
#extractPattern(myTrends, Pattern = c("up", "down"), Delay = 20)
```

---

fitSegBIC *Fit segmented regression models on a feature/gene*

Description

fits segmented regression models

Usage

```
fitSegBIC(Data, maxK = 2, tVectIn = NULL, minNumInSeg = 5,
  pvalCut = 0.1, numTry = 5, keepFit = FALSE)
```

Arguments

Data	a matrix of normalized expression measurements. Rows are genes/features and columns are samples.
maxK	maximum number of breakpoints to consider. For each gene, trendy will fit maxK + 1 models containing 0 -> maxK breakpoints (1 -> (maxK + 1) segments). The model with the lowest BIC value will be selected (unless forceRsq = TRUE, see below).

tVectIn	a numerical vector indicating the time-points or the order of samples. If it is NULL (default), then the time/order will be assumed to be equaly spaced from 1:N (N is number of samples).
minNumInSeg	minimum number of samples required to be within a segment. If a breakpoint model has a segment with fewer than minNumInSeg point in any segment, then the model is not considered valid.
pvalCut	p-value cutoff. If the p-value of a segment is greater than PvalCut, then the segment will be called as 'no change'.
numTry	the number of different seeds to try. If all numTry runs fail, then the linear regression (no breakpoints, one segment) model will be returned.
keepFit	whether to report the fitted object (default is FALSE).

Value

Trend: direction of each sample; -1: down, 0: no change, 1: up Slope: fitted slopes, Slope.Trend: sign of fitted slopes, Slope.Pvalue: p value of each segment, Breakpoint: estimated breakpoints, Fitted.Values: fitted values AdjustedR2: adjusted r value of the model Fit: fit object

Author(s)

Rhonda Bacher and Ning Leng

---

formatFunc *internal helper function to format results*

Description

helper function to format result

Usage

```
formatFunc(IN)
```

Arguments

IN	the object to be formatted

Value

a formated matrix of results

Author(s)

Rhonda Bacher

formatResults *Function to format results for saving.*

Description

format data from Trendy which can be saved for later use.

Usage

```
formatResults(topTrendyData, featureNames = NULL)
```

Arguments

topTrendyData	results from topTrendy() function
featureNames	an optional vector of features (if only interested in outputting a subset of features/genes).

Value

The function will reformat the output from Trendy so that it can be easily save as a .txt or .csv file. If featureNames is supplied then only the information for those features/genes is returned.

Author(s)

Rhonda Bacher

Examples

```
data(trendyExampleData)
 myTrends <- trendy(Data=trendyExampleData[seq_len(2),])
 myTrends <- results(myTrends)
 topTrendyRes <- topTrendy(myTrends)
 resToSave <- formatResults(topTrendyRes)
```

getCounts *getCounts*

Description

Convenient helper function to extract the normalized expression matrix from the SummarizedExperiment

Usage

```
getCounts(DATA)
```

Arguments

DATA	An object of class SummarizedExperiment that contains expression data and metadata

Value

A matrix which contains the expression data where genes/features are in rows and samples are in columns

Examples

```
m1 <- matrix(c(c(rnorm(50,5,1),sort(rnorm(50, 15, 5))), rnorm(100, 50,10)), 2, 100, TRUE)
ExampleData <-
SummarizedExperiment::SummarizedExperiment(assays=list("Counts"=m1))
myData <- getCounts(ExampleData)
```

---

plotFeature *Plot features of interest*

Description

plot each feature with (or without) the fitted trend.

Usage

```
plotFeature(Data, tVectIn = NULL, featureNames, showFit = TRUE,
  simple = FALSE, showLegend = TRUE, trendyOutData = NULL,
  cexLegend = 1, legendLocation = "side", xlab = "Time",
  ylab = "Normalized Expression", segColors = c("chartreuse3",
  "coral1", "black", "cornflowerblue"), customTitle = NULL,
  customLabels.x = NULL, spacing.x = NULL)
```

Arguments

Data	a matrix of normalized expression measurements. Rows are genes/features and columns are samples.
tVectIn	a numerical vector indicating the time-points or the order of samples. If it is NULL (default), then the time/order will be assumed to be equaly spaced from 1:N (N is number of samples).
featureNames	a list of genes or features to plot
showFit	whether to plot the segmented regression fitting (default is TRUE)
simple	if TRUE the plot will not highlight the breakpoints and segments and will only display a black fitted line. (default is FALSE)
showLegend	if TRUE and simple=FALSE then a legend will be output (default = TRUE)

trendyOutData	segmented regression fitting result from running trendy(); if showFit is TRUE and trendyOutData is NULL, then the segmented regression will be fit for each of the genes and it may take longer to run
cexLegend	cex option for sizing of legend text, default is 1.
legendLocation	whether to place the legend to the right 'side' of each plot or at the 'bottom' of a multo-panelled plot (default is 'side').
xlab	x-axis name
ylab	y-axis name
segColors	define colors for the 'breakpoint', and 'up', 'same', 'down' segments (default: segColors = c("chartreuse3", "coral1", "black", "cornflowerblue"))
customTitle	default is set the plot title as the name of the feature. Otherwise this should be a named vector, with the featureName as the name and the element as the desired plot title. (i.e. customTitle <- c("MyTitle" = gene1)).
customLabels.x	specify x-axis tick labels instead of using the default values from tVectIn.
spacing.x	specify x-axis tick spacing, smaller values give more tick marks.

Value

plot of gene expression and fitted line

Author(s)

Ning Leng and Rhonda Bacher

Examples

```
d1 <- matrix(c(c(rnorm(50,5,1),sort(rnorm(50, 15, 5))), rnorm(100, 50,10)), 2, 100, TRUE)
rownames(d1) <- c("g1","g2")
colnames(d1) <- paste0("time", seq_len(100))
plotFeature(d1, featureNames=c("g1","g2"))
```

---

results *results*

Description

Convenient helper function to extract the results of running Trendy. Results data.frames/matrices are stored in the `metadata` slot and can also be accessed without the help of this convenience function by calling `metadata()`.

Usage

```
results(DATA, type = c("TrendyFits"))
```

Arguments

DATA	An object of class SummarizedExperiment that contains normalized expression and other metadata, and the output of the Trendy function.
type	A character variable specifying which output is desired, with possible values "TrendyFits". By default results() will return type="TrendyFits", which is the matrix of normalized counts from SCnorm.

Value

A data.frame containing output as detailed in the description of the type input parameter

Examples

```
data(trendyExampleData)
Conditions = rep(c(1), each= 90)
trendyOut <- trendy(Data=trendyExampleData[seq_len(2),])
trendyResults <- results(trendyOut)
```

---

topTrendy *obtain top genes from trendy results*

Description

reformats the list output for genes with a given adjusted R^2 cutoff

Usage

```
topTrendy(trendyOutData, adjR2Cut = 0.5)
```

Arguments

trendyOutData	output from the trendy function
adjR2Cut	cutoff for the adjusted R^2. Genes whose adjusted R^2 is greater than adjR2Cut are called as significant.

Value

only significant genes will be included in the output. The output is reformatted as: Trend direction of each sample; -1: down, 0: no change, 1: up Slope: fitted slopes, Slope.Trend: sign of fitted slopes, Slope.Pvalue: p value of each segment, Breakpoint: estimated breakpoints, Fitted.Values: fitted values AdjustedR2: adjusted r value of the model Fit: fit object

Examples

```
d1 <- matrix(c(c(rnorm(50,5,1),sort(rnorm(50, 15, 5))), rnorm(100, 50,10)), 2, 100, TRUE)
rownames(d1) <- c("g1","g2")
colnames(d1) <- paste0("time", seq_len(100))
seg.all <- trendy(d1)
seg.all <- results(seg.all)
top.genes <- topTrendy(seg.all)
```

`trendHeatmap` *Draw heatmap of gene expression trends*

Description

heatmap of the fitted trends

Usage

```
trendHeatmap(topTrendyData, featureNames = NULL, cexRow = 0.5,
  cexCol = 0.5)
```

Arguments

`topTrendyData`	results from topTrendy() function.
`featureNames`	names of features/genes to plot if the heatmap should be restricted. Deafult is to plot all genes from topTrendy() function.
`cexRow`	relative text size of row labels, default=.5.
`cexCol`	relative text size of column labels, default=.5.

Value

The function takes significant genes/features called from the topTrendyData() function. These genes are further grouped into three groups: up, down, or no change in the first segment. Within each group, the genes are sorted by their first break point. The heatmap shows expression trends of these three groups of genes. In the heatmap, red/blue/black represents up/down/nochange. A list of genes in the heatmap order is returned.

Author(s)

Ning Leng and Rhonda Bacher

Examples

```
m1 <- matrix(c(c(rnorm(50,5,1),sort(rnorm(50, 15, 5))), rnorm(100, 50,10)), 2, 100, TRUE)
rownames(m1) <- c("g1","g2")
colnames(m1) <- paste0("time", seq_len(100))
myTrends <- results(trendy(m1))
topGenes <- topTrendy(myTrends)
#makeHeat <- trendHeatmap(topGenes)
```

trendy *Trendy*

Description

Segmented regression models are fit for each gene. The number of model fits is 1 -> maxK.

Usage

```
trendy(Data = NULL, tVectIn = NULL, saveObject = FALSE,
  fileName = NULL, meanCut = 10, maxK = 3, minNumInSeg = 5,
  pvalCut = 0.1, numTry = 5, keepFit = FALSE, NCores = NULL,
  featureNames = NULL)
```

Arguments

Data	a matrix of normalized expression measurements. Rows are genes/features and columns are samples.
tVectIn	a numerical vector indicating the time-points or the order of samples. If it is NULL (default), then the time/order will be assumed to be equaly spaced from 1:N (N is number of samples).
saveObject	if TRUE then the trendy object produced will be saved to use in the Shiny app (default is FALSE).
fileName	the file name (and file path) to save the Trendy object, only used if saveObject=TRUE (default name is trendyOutputForShiny.RData).
meanCut	genes whose mean is less than MeanCut will not be considered, default is 10.
maxK	maximum number of breakpoints to consider. For each gene, trendy will fit maxK + 1 models containing 0 -> maxK breakpoints (1 -> (maxK + 1) segments). The model with the lowest BIC value will be selected (unless forceRsq = TRUE, see below).
minNumInSeg	minimum number of samples required to be within a segment. If a breakpoint model has a segment with fewer than minNumInSeg point in any segment, then the model is not considered valid.
pvalCut	p-value cutoff. If the p-value of a segment is greater than PvalCut, then the segment will be called as 'no change'.
numTry	the number of different seeds to try. If all numTry runs fail, then the linear regression (no breakpoints, one segment) model will be returned.
keepFit	whether to report the fitted object (default is FALSE).
NCores	number of cores to use, default is detectCores() - 1.
featureNames	optional parameter to specify an explicit subset of features/genes to fit the segmented regression model to.

Value

Trend: direction of each sample; -1: down, 0: no change, 1: up Slope: fitted slopes, Slope.Trend: sign of fitted slopes, Slope.Pvalue: p value of each segment, Breakpoint: estimated breakpoints, Fitted.Values: fitted values AdjustedR2: adjusted R squared value of the model Fit: fit object

Author(s)

Ning Leng and Rhonda Bacher

Examples

```
m1 <- matrix(c(c(rnorm(50,5,1),sort(rnorm(50, 15, 5))), rnorm(100, 50,10)), 2, 100, TRUE)
rownames(m1) <- c("g1","g2")
colnames(m1) <- paste0("time", seq_len(100))
myTrends <- trendy(m1)
```

---

trendyExampleData *Example dataset for Trendy*

Description

Example time-course dataset.

Usage

```
data(trendyExampleData)
```

Format

data matrix

Examples

```
data(trendyExampleData)
```

---

trendyShiny *Trendy shiny app to interactively vizualize results after running trendy().*

Description

Trendy shiny app to interactively vizualize results after running trendy().

Value

Opens a browser window with an interactive `shiny` app and visualize all precomputed Trendy fits.

Index