Package 'Trendy'

August 21, 2017

Type Package

Title Breakpoint analysis of time course gene expression data

Version 0.99.0

Author Rhonda Bacher and Ning Leng

Maintainer Rhonda Bacher <rbacher@wisc.edu>

Description Trendy implements segmented regression to detect breakpoints for each gene in high throughput data with ordered conditions.

Imports stats, graphics, grDevices, segmented, gplots, parallel

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Suggests knitr

VignetteBuilder knitr

NeedsCompilation no

R topics documented:

bpdist *Distribution of breakpoints*

Description

calculates number of breakpoints at each time.

Usage

```
bpdist(Top.Trendy, N.Digits = 0)
```

Arguments

Top.Trendy	results from toptrendy() function
N.Digits	how many digits to be used when rounding (default is 0 (return integers))

Value

The function takes significant genes called from the toptrendy() 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

extractpattern *Extract pattern from segmented regression*

Description

find dynamic genes that follow a given pattern

Usage

```
extractpattern(Seg.Data, Pattern = NULL, Radj.Cut = 0.5, Delay = 0)
```

Arguments

Seg.Data	output from trendy() function
Pattern	vector containing pattern to search genes/features. If length is one then it will only consider features with constant pattern across the entire time-course.
Radj.Cut	only consider features with adjusted R^2 > Radj.Cut. Default = .5.
Delay	search for pattern starting after certain time-point (e.g. only genes with a breakpoint > 10).

Value

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

Author(s)

Rhonda Bacher

Examples

```
myTrends <- trendy(TrendyExData)
extractpattern(myTrends, Pattern = c("up")) #increasing only features
 extractpattern(myTrends, Pattern = c("up", "down")) #features with a peak
 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, Max.K = 5, T.Vect = NULL, Min.Num.In.Seg = 5,
  Pval.Cut = 0.1, Num.Try = 100, Keep.Fit = FALSE)
```

Arguments

Data	matrix of normalized expression measurements. Rows are genes and columns are samples.
Max.K	maximum number of breakpoints to consider. For each gene, the trendy will fit Max.K + 1 models containing 0 -> Max.K breakpoints (1 -> (Max.K+1) segments). The model with the highest adjusted r^2 value will be selected.
T.Vect	a numerical vector indicating the time points. If it is NULL (default), then the time will be assumed to be equaly spaced from 1:N (N is number of samples).
Min.Num.In.Seg	minimum number of samples required to be within a segment.
Pval.Cut	p-value cutoff. If the p-value of a segment is greater than PvalCut, then the segment will be called as 'no change'.
Num.Try	the number of different seeds to try. If all NumTry runs fail, then linear regression (no breakpoints, one segment) model will be returned.
Keep.Fit	whether report the fitted object (default is FALSE).

Value

id.sign: direction of each sample; -1: down, 0: no change, 1: up slp: fitted slopes, slp.sign: sign of fitted slopes, slp.pval: p value of each segment, bp: estimated breakpoints, fitted: fitted values radj: adjusted r value of the model fit: fit object

Author(s)

Ning Leng and Rhonda Bacher

Examples

```
d1 <- rbind(c(rep(1,50),1:50), c(100:1))
rownames(d1) <- c("g1","g2")
fit.seg(d1, "g1")
```

`fitsegRsq` *Fit Segmented regression models on a feature/gene*

Description

fits segmented regression models

Usage

```
fitsegRsq(Data, T.Vect = NULL, Max.K = 5, Min.Num.In.Seg = 5,
  Pval.Cut = 0.1, Cut.Diff = 0.1, Num.Try = 100, Keep.Fit = FALSE)
```

Arguments

`Data`	matrix of normalized expression measurements. Rows are genes and columns are samples.
`T.Vect`	a numerical vector indicating the time points. If it is NULL (default), then the time will be assumed to be equaly spaced from 1:N (N is number of samples).
`Max.K`	maximum number of breakpoints to consider. For each gene, the trendy will fit Max.K + 1 models containing 0 -> Max.K breakpoints (1 -> (Max.K+1) segments). The model with the highest adjusted r^2 value will be selected.
`Min.Num.In.Seg`	minimum number of samples required to be within a segment.
`Pval.Cut`	p-value cutoff. If the p-value of a segment is greater than PvalCut, then the segment will be called as 'no change'.
`Cut.Diff`	Only used if Force.Rsq is set to TRUE. If the difference between the r^2 from the k + 1 breakpoint model and the r^2 from the k breakpoint model is less than CutFiff, then the optimal number of breakpoints is set as k instead of k + 1.
`Num.Try`	the number of different seeds to try. If all NumTry runs fail, then linear regression (no breakpoints, one segment) model will be returned.
`Keep.Fit`	whether report the fitted object (default is FALSE).

Value

id.sign: direction of each sample; -1: down, 0: no change, 1: up slp: fitted slopes, slp.sign: sign of fitted slopes, slp.pval: p value of each segment, bp: estimated breakpoints, fitted: fitted values radj: adjusted r value of the model fit: fit object

Author(s)

Ning Leng and Rhonda Bacher

Examples

```
d1 <- rbind(c(rep(1,50),1:50), c(100:1))
rownames(d1) <- c("g1","g2")
fitseg(d1, "g1")
```

formatfunc *helper function to format results*

Description

helper function to format result

Usage

```
formatfunc(IN, NAME)
```

Arguments

IN	the object to be formatted
NAME	the name of the object

Value

a formated matrix of results

Author(s)

Rhonda Bacher

formatresults *Function to format results for saving.*

Description

format data to save.

Usage

```
formatresults(Top.Trendy, Feature.Names = NULL)
```

Arguments

Top.Trendy	results from toptrendy() function
Feature.Names	an optional vector of genes to use.

Value

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

Author(s)

Rhonda Bacher

Orig.Data *Example dataset for Trendy Shiny App*

Description

Example of output from running trendy() on TrendyExData.RData (see vignette)

Usage

```
data(exampleObject_trendyForShiny)
```

Format

list

Examples

```
data(exampleObject_trendyForShiny)
```

plotfeature *Plot features of interest*

Description

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

Usage

```
plotfeature(Data, T.Vect = NULL, Feature.Names, Seg.Fit = TRUE,
  Seg.Data = NULL, Cond.Col = NULL, Cond.Col.Sample = NULL,
  Cond.Prefix = "Day", Y.Name = "Normalized Expression", X.Name = "Time")
```

Arguments

Data	matrix of normalized expression measurements. Rows are genes and columns are samples.
T.Vect	a numerical vector indicating the time points. If it is NULL (default), then the time will be assumed to be equaly spaced from 1:N (N is number of samples).
Feature.Names	a list of genes or features to plot
Seg.Fit	whether plot the segmented regression fitting (default is TRUE)
Seg.Data	segmented regression fitting result from running trendy(); i f Seg.Data is NULL and Seg.Fit = T, then the segmented regression will be fit for each of the genes and it may take longer to run
Cond.Col	color for each condition, names of this vector should be condition names; if it is NULL (default), no legend will be generated
Cond.Col.Sample	each sample's color. The vector's length should match number of samples. if it is NULL, expression will be shown in black (default).
Y.Name	y-axis name
X.Name	x-axis name

Value

plot of gene expression and fitted line

Author(s)

Ning Leng and Rhonda Bacher

Examples

```
d1 <- rbind(c(rep(1,50),1:50), c(100:1))
rownames(d1) <- c("g1","g2")
plotmarker(d1, Feature.Names=c("g1","g2"))
```

`toptrendy` *obtain top genes from trendy results*

Description

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

Usage

```
toptrendy(Seg.Data, AdjR.Cut = 0.5)
```

Arguments

`Seg.Data`	output from the trendy function
`AdjR.Cut`	cutoff for the adjusted r^2. Genes whose adjusted r^2 is greater than adjR.Cut are called as significant.

Value

only significant genes will be included in the output. The output is reformatted as: id.sign direction of each sample; -1: down, 0: no change, 1: up slp: fitted slopes, slp.sign: sign of fitted slopes, slp.pval: p value of each segment, bp: estimated breakpoints, fitted: fitted values radj: adjusted r value of the model fit: fit object

Examples

```
d1 <- rbind(c(rep(1,50),1:50), c(100:1))
     rownames(d1) <- c("g1","g2")
     seg.all <- trendy(d1)
top.seg <- toptrendy(seg.all)
```

trendheatmap *Draw heatmap of gene expression trends*

Description

heatmap of the fitted trends

Usage

```
trendheatmap(Top.Trendy)
```

Arguments

Top.Trendy	results from toptrendy() function

Value

The function takes significant genes/features called from the toptrendy() 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

trendy *segmented regression on a set of genes*

Description

Segmented regression models are fit for each genes. The number of model fits is 1 -> Max.K.

Usage

```
trendy(Data = NULL, T.Vect = NULL, Save.Object = FALSE,
  File.Name = NULL, Mean.Cut = 10, Max.K = 3, Min.Num.In.Seg = 5,
  Pval.Cut = 0.1, Cut.Diff = 0.1, Num.Try = 100, Keep.Fit = FALSE,
  Force.Rsq = FALSE, N.Cores = NULL)
```

Arguments

Data	matrix of normalized expression measurements. Rows are genes and columns are samples.
T.Vect	a numerical vector indicating the time points. If it is NULL (default), then the time will be assumed to be equaly spaced from 1:N (N is number of samples).
Save.Object	If TRUE then the trendy object produced will be saved to use in the shiny app (default is FALSE).

File.Name	the file name (and file path) to save the Trendy object, only used if SaveObject=TRUE.
Mean.Cut	genes whose mean is less than MeanCut will not be considered, default is 10.
Max.K	maximum number of breakpoints to consider. For each gene, the trendy will fit Max.K + 1 models containing 0 -> Max.K breakpoints (1 -> (Max.K+1) segments). The model with the highest adjusted r^2 value will be selected.
Min.Num.In.Seg	minimum number of samples required to be within a segment.
Pval.Cut	p-value cutoff. If the p-value of a segment is greater than PvalCut, then the segment will be called as 'no change'.
Cut.Diff	Only used if Force.Rsq is set to TRUE. If the difference between the r^2 from the k + 1 breakpoint model and the r^2 from the k breakpoint model is less than CutFiff, then the optimal number of breakpoints is set as k instead of k + 1.
Num.Try	the number of different seeds to try. If all NumTry runs fail, then linear regression (no breakpoints, one segment) model will be returned.
Keep.Fit	whether report the fitted object (default is FALSE).
Force.Rsq	whether to use adjusted r^2 in CutDiff evaluation instead of BIC, corresponds to very early versions of Trendy.
N.Cores	number of cores to use, default is detectCores() - 1.

Value

id.sign: direction of each sample; -1: down, 0: no change, 1: up slp: fitted slopes, slp.sign: sign of fitted slopes, slp.pval: p value of each segment, bp: estimated breakpoints, fitted: fitted values radj: adjusted r value of the model fit: fit object

Author(s)

Ning Leng and Rhonda Bacher

Examples

```
d1 <- rbind(c(rep(1,50),1:50), c(100:1))
     rownames(d1) <- c("g1","g2")
     trendy(d1)
```

TrendyExData *Example dataset for Trendy*

Description

Example time-course dataset.

Usage

```
data(TrendyExData)
```

Format

data matrix

Examples

```
data(TrendyExData)
```

Index