Package ‘OGSA’
April 26, 2017

Version 1.6.0
Date 2015-06-25
Title Outlier Gene Set Analysis
Author Michael F. Ochs <ochsm@tcnj.edu>
Description OGSA provides a global estimate of pathway deregulation in
cancer subtypes by integrating the estimates of significance
for individual pathway members that have been identified by
outlier analysis.
Maintainer Michael F. Ochs <ochsm@tcnj.edu>
Depends R (>= 3.2.0)
Imports gplots(>= 2.8.0), limma(>= 3.18.13), Biobase
LazyData true
License GPL (== 2)
biocViews GeneExpression, Microarray, CopyNumberVariation
NeedsCompilation no

R topics documented:

OGSA-package ......................................................... 2
cnv ................................................................. 2
copaInt ............................................................ 3
copaIntE ............................................................ 4
copaStat ............................................................ 5
expr ................................................................. 6
meth ................................................................. 6
outCallRank ......................................................... 6
outCallRankE ......................................................... 7
outCallTib ........................................................... 9
outCallTibE .......................................................... 10
outCount ........................................................... 11
outMap .............................................................. 12
outRank ............................................................ 13
pathGS .............................................................. 14
pheno ............................................................... 14
testGScogps ....................................................... 15

Index 16
OGSA-package

This package uses outlier statistics and gene set analysis to identify deregulated pathways.

Description

The package applies three versions of outlier statistics across multiple molecular data types to create a single estimate at the gene level for the number of outliers relative to normal controls. These gene estimates are used in gene set analysis to determine deregulated pathways. Visualization of outlier calls provide sample specific information on potential drivers of the gene set statistic.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>OGSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.0</td>
</tr>
<tr>
<td>Date:</td>
<td>2015-01-01</td>
</tr>
<tr>
<td>License:</td>
<td>Gnu Public License</td>
</tr>
</tbody>
</table>

Author(s)

Michael Ochs

Maintainer: Michael Ochs <ochsm@tcnj.edu>

References


cnv

Copy number variation data

Description

Matrix of copy number variation data.

Usage

```r
cnv
```

Format

Matrix of 2000 rows by 69 columns with copy number variation data
Description
Counts outliers by Tibshirani-Hastie method by calling outCount after setting up list or by rank outlier method by calling outRank

Usage
copaInt(dataSet, phenotype, tails, thres = 0.05, method='Tibshirani', corr=FALSE, offsets=NULL)

Arguments
dataset: Set of matrices of molecular data
phenotype: Vector of 1 for case, 0 for control
tails: Vector equal to number of matrices with values left or right for where to find outliers
thres: alpha value
method: Tibshirani, Rank
corr: Whether to correct for normal outliers
offsets: A vector equal to the number of matrices which sets the minimum value relative to normal to call outlier (corrected rank only)

Value
A vector with outlier counts by gene

References

Examples
data(ExampleData)

# Set up phenotype
phenotype <- pheno
names(phenotype) <- colnames(cnv)

tailLRL <- c('left', 'right', 'left')

dataSet <- list(expr, meth, cnv)
tibLRL <- copaInt(dataSet, phenotype, tails=tailLRL)
copaIntE  copaIntE

Description
Counts outliers by Tibshirani-Hastie method by calling outCount after setting up list or by rank outlier method by calling outRank

Usage
```r
copaIntE(expressionSet, tails, thres = 0.05, method='Tibshirani', corr=FALSE, offsets=NULL)
```

Arguments
- `expressionSet`: object containing Set of matrices of molecular data and phenotype data (1 for case, 0 for control)
- `tails`: Vector equal to number of matrices with values left or right for where to find outliers
- `thres`: alpha value
- `method`: Tibshirani, Rank
- `corr`: Whether to correct for normal outliers
- `offsets`: A vector equal to the number of matrices which sets the minimum value relative to normal to call outlier (corrected rank only)

Value
A vector with outlier counts by gene

References

Examples
```r
data(ExampleData)
library(Biobase)
# building the Annotated Data Frame
phenoData <- AnnotatedDataFrame(
  data.frame(
    type = factor(x = pheno, labels = c("Control", "Case")),
    row.names = colnames(expr)
  )
)
# build environment
inputData <- list2env(list(exprs = expr, meth = meth, cnv = cnv))
# build expressionSet - other information can be added here
expressionSet <- ExpressionSet(inputData, phenoData)

# set up values for expr-meth-cnv in that order
tailLRL <- c("left", "right", "left")

tibLRL <- copaIntE(expressionSet, tails=tailLRL)

Description
Calculates outlier statistics by the Tibshirani-Hastie method

Usage

\[
\text{copaStat (data, phenotype, tail='right', perms=100, permType='array')}
\]

Arguments

data A matrix of nGene by nSample
phenotype A vector of 0s and 1s of length nSample, where 1 = case, 0 = control
tail Indicates whether outliers are up (right) or down (left) outliers
perms The number of permutations
permType By all on array or by gene, if by gene increase perms significantly and plan on lots of time; in theory array should be fine as genes are rescaled

Value
A vector with outlier counts by gene

References

Examples

data(ExampleData)

#Set up phenotype
phenotype <- pheno	names(phenotype) <- colnames(cnv)

#set up values for expr-meth-cnv in that order
tailLRL <- c("left", "right", "left")

#setup dataList
dataSet <- list(expr, meth, cnv)
```r
data <- dataSet[[1]]
tibL <- copaStat(data, phenotype, tail='right', perms=100, permType='array')
```

### Description

**expr**

Matrix of expression data

**Usage**

```r
expr
```

**Format**

Matrix of 2000 rows by 69 columns with expression data

### Description

**meth**

Matrix of methylation data

**Usage**

```r
meth
```

**Format**

Matrix of 2000 rows by 69 columns with methylation data

### Description

**outCallRank**

Counts outliers by the Ghosh method and generates list objects with all outliers noted

**Usage**

```r
outCallRank (dataSet, phenotype, thres= 0.05, tail='right', corr=FALSE, offsets=NULL, names=NULL)
```
outCallRankE

Arguments

- `dataSet`: Set of matrices of molecular data
- `phenotype`: A vector of 0s and 1s of length nSample, where 1 = case, 0 = control
- `thres`: Alpha value
- `tail`: A vector equal to the number of matrices with values left or right for where to find outliers
- `corr`: Whether to correct for normal outliers
- `offsets`: A vector equal to the number of matrices which sets the minimum value relative to normal to call outlier (corrected rank only)
- `names`: A vector equal to the number of matrices to name molecular type of data (e.g., CNV)

Value

A list with all specific outlier calls for each molecular type in each case sample

References


Examples

data(ExampleData)

# set up dataSet
dataSet <- list(expr, meth,cnv)

# Set up Phenotype
phenotype <- pheno
names(phenotype) <- colnames(cnv)

# set up values for expr-meth-cnv in that order
taxilRL <- c('left', 'right', 'left')

outRankLRL <- outCallRank(dataSet, phenotype, names=c('Expr', 'Meth', 'CNV'), tail=taxilRL)

Description

Counts outliers by the Ghosh method and generates list objects with all outliers noted
Usage

```r
callRankE (expressionSet, thres= 0.05, tail='right', corr=FALSE,
offsets=NULL, names=NULL)
```

Arguments

- `expressionSet`: object containing Set of matrices of molecular data and phenotype data (1 for case, 0 for control)
- `thres`: Alpha value
- `tail`: A vector equal to the number of matrices with values left or right for where to find outliers
- `corr`: Whether to correct for normal outliers
- `offsets`: A vector equal to the number of matrices which sets the minimum value relative to normal to call outlier (corrected rank only)
- `names`: A vector equal to the number of matrices to name molecular type of data (e.g., CNV)

Value

A list with all specific outlier calls for each molecular type in each case sample

References


Examples

```r
data(ExampleData)

library(Biobase)
# building the Annotated Data Frame
phenoData <- AnnotatedDataFrame(
  data.frame(
    type = factor(x = pheno, labels = c("Control", "Case")),
    row.names = colnames(expr)
  )
)
# build environment
inputData <- list2env(list(exprs = expr, meth = meth, cnv = cnv))
# build expressionSet - other information can be added here
expressionSet <- ExpressionSet(inputData, phenoData)
# set up values for for the tails in the order that they are exported,
# for example:
tailLRL <- c('left', 'right', 'left')
outRankLRL <- callRankE(expressionSet, names=c('Expr', 'Meth', 'CNV'),
tail=tailLRL)
```
Description
Counts outliers by the Tibshirani and Hastie method and generates a list object with all outliers noted.

Usage
outCallTib (dataSet, phenotype, tail='right', corr=FALSE, names=NULL)

Arguments
dataSet Set of matrices of molecular data
phenotype A vector of 0s and 1s of length nSample, where 1 = case, 0 = control
tail Vector equal to number of matrices with values 'left' or 'right' for where to find outliers
corr whether to correct for normal outliers ONLY for compatibility, since method does not allow determining specific changes in cases, it will just print message if corr = TRUE
names Vector equal to number of matrices to name molecular type of data (e.g., 'CNV').

Value
A list with all specific outlier calls for each molecular type in each case sample.

References

Examples
data(ExampleData)
data('KEGG_BC_GS')

# Set up dataSet
dataSet <- list(expr, meth, cnv)

# Set up Phenotype
phenotype <- pheno
names(phenotype) <- colnames(cnv)

# set up values for expr-meth-cnv in that order
tailLRL <- c('left', 'right', 'left')

outTibLRL <- outCallTib(dataSet, phenotype, names=c('Expr', 'Meth', 'CNV'), tail=tailLRL)
Description
Counts outliers by the Tibshirani and Hastie method and generates a list object with all outliers noted.

Usage
outCallTibE(expressionSet, tail='right', corr=FALSE, names=NULL)

Arguments
expressionSet ExpressionSet object containing sets of data and phenotype information
tail Vector equal to number of matrices with values 'left' or 'right' for where to find outliers
corr whether to correct for normal outliers ONLY for compatibility, since method does not allow determining specific changes in cases, it will just print message if corr = TRUE
names Vector equal to number of matrices to name molecular type of data (e.g., 'CNV').

Value
A list with all specific outlier calls for each molecular type in each case sample.

References

Examples
data(ExampleData)
data('KEGG_BC_GS')
library(Biobase)
# building the Annotated Data Frame
phenoData <- AnnotatedDataFrame(
    data.frame(  
        type = factor(x = pheno, labels = c("Control", "Case")),  
        row.names = colnames(expr)
    )
)
# build environment
inputData <- list2env(list(exprs = expr, meth = meth, cnv = cnv))
# build expressionSet - other information can be added here
expressionSet <- ExpressionSet(inputData, phenoData)
outCount

# set up values for for the tails in the order that they are exported, for example:
tailLRL <- c('left', 'right', 'left')

toutTibLRL <- outCallTibE(expressionSet, names=c('Expr', 'Meth', 'CNV'), tail=tailLRL)

Description
Counts outliers by the Tibshirani and Hastie method. Adds the ability to subtract for outliers in the normals using corr = TRUE

Usage
outCount (data, phenotype, tail='right', corr=FALSE)

Arguments
data A matrix of nGene by nSample
phenotype A vector of 0s and 1s of length nSample, where 1 = case, 0 = control
tail Indicates whether outliers are up (right) or down (left) outliers
corr Whether to correct for normal outliers

Value
A vector with outlier counts by gene

References

Examples
data(ExampleData)
# Set up Phenotype
phenotype <- pheno
names(phenotype) <- colnames(cnv)

#set up datalist
dataSet <- list(expr,meth,cnv)

# set up values for expr-meth-cnv in that order
tailLRL <- c('left', 'right', 'left')

outTibLRL <- outCallTib(dataSet, phenotype=pheno,
                        names=c('Expr', 'Meth', 'CNV'), tail=tailLRL)
outMap

Description

Creates PDF color map of where outliers occur coded for molecular type

Usage

outMap (outList, geneList, hmName = 'PatSpecMap.pdf', plotName = 'Outliers', truncGene = FALSE, clust=FALSE)

Arguments

outList     List with all outliers generated by outCallRank or outCallTib
geneList    Gene set to compare against
hmName      Name for PDF output file
plotName    Header for plot
truncGene   if TRUE, only include genes that have outlier in the plot, default is all genes in
gene set
clust       If TRUE, clusters data and produces dendrograms

Value

A matrix used for generating heatmap

References

Analysis and Top Scoring Pair for Integrated Data Analysis and Biomarker Discovery. IEEE/ACM
Transactions on Computational Biology and Bioinformatics, 1-1. doi:10.1109/tcbb.2013.153

Examples

data(ExampleData)
data('KEGG_BC_GS')

# Set up Phenotype
phenotype <- pheno
names(phenotype) <- colnames(cnv)

# Set up datalist
dataSet <- list(expr,meth,cnv)

# set up values for expr-meth-cnv in that order
tailLRL <- c('left', 'right', 'left')

outTibLRL <- outCallTib(dataSet, phenotype=pheno,
                        names=c('Expr', 'Meth', 'CNV'), tail=tailLRL)

# put in your pathways here
\textbf{outRank}

\begin{verbatim}
pdgfB <- pathGS$'BIOCARTA_PDGF_PATHWAY'
outMap(outTibLRL, pdgfB, hmName='BC_PDGF_TIB.pdf', plotName='PDGF
Outlier T-H LRL Calls')
\end{verbatim}

\textbf{Description}

Counts outliers by the Ghosh method.

\textbf{Usage}

\begin{verbatim}
outRank (dataSet, phenotype, thres= 0.05, tail='right', corr=FALSE,
offsets=NULL)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
\item \textbf{dataSet} Set of matrices of molecular data
\item \textbf{phenotype} Vector of 1 for case, 0 for control
\item \textbf{thres} Alpha value
\item \textbf{tail} Vector equal to number of matrices with values 'left' or 'right' for where to find outliers
\item \textbf{corr} Whether to correct for normal outliers
\item \textbf{offsets} Vector equal to number of matrices which sets minimum value relative to normal to call outlier (corrected rank only)
\end{itemize}

\textbf{Value}

A vector with outlier counts by gene

\textbf{References}


\textbf{Examples}

\begin{verbatim}
data(ExampleData)

# Set up Phenotype
phenotype <- pheno
names(phenotype) <- colnames(cnv)

#set up dataSet
dataSet <- list(expr, meth, cnv)

# set up values for expr-meth-cn in that order
\end{verbatim}
tailLRL <- c('left', 'right', 'left')

outRankLRL <- outRank(dataSet, phenotype, thres= 0.05, tail=tailLRL,
corr=FALSE, offsets=NULL)

---

pathGS  
*Gene set defined by BioCarta pathway*

**Description**
A large list containing gene set from the BioCarta pathway

**Usage**

pathGS

**Format**

List of 403 elements

**Details**

Contained in KEGG_BC_GS data frame

**Source**


---

pheno  
*pheno*

**Description**
Vector of phenotype data

**Usage**

pheno

**Format**

A vector of 0s and 1s of length 69, where 1 = tumor, 0 = normal
Description

Performs gene set test on outlier counts

Usage

testGScogps (outlierCts, geneSets)

Arguments

outlierCts Vector with gene names and outlier counts
geneSets List of gene sets

Value

A vector with rank sum gene set statistics

References


Examples

## Not run:
data(ExampleData)
data('BC_GS')

#Set up your phenotype
phenotype <- rep(0, 69)
phenotype[annot[, 3] == 'Event'] <- 1
names(phenotype) <- rownames(annot)

# set up values for expr-meth-cnv in that order
tailLRL <- c('left', 'right', 'left')

dataSet <- list(expr, meth, cnv)
tibLRLcorr <- copaInt(dataSet, phenotype, tails=tailLRL, corr=TRUE)
gsTibLRLcorr <- testGScogps(tibLRLcorr, pathGS)

## End(Not run)
Index

cnv, 2

copaInt, 3
copaIntE, 4
copaStat, 5

expr, 6

meth, 6

OGSA (OGSA-package), 2
OGSA-package, 2
outCallRank, 6
outCallRankE, 7
outCallTib, 9
outCallTibE, 10
outCount, 11
outMap, 12
outRank, 13

pathGS, 14
pheno, 14

testGScogps, 15