Package 'multiWGCNA'

January 18, 2025

Type Package

Title multiWGCNA

Version 1.5.0

Description An R package for deeping mining gene co-expression networks in multi-trait expression data. Provides functions for analyzing, comparing, and visualizing WGCNA networks across conditions. multiWGCNA was designed to handle the common case where there are multiple biologically meaningful sample traits, such as disease vs wildtype across development or anatomical region.

License GPL-3

Encoding UTF-8

LazyData true

Suggests BiocStyle, doParallel, ExperimentHub, knitr, markdown, rmarkdown, testthat (>= 3.0.0), vegan

VignetteBuilder knitr

RoxygenNote 7.2.3

biocViews Sequencing, RNASeq, GeneExpression, DifferentialExpression, Regression, Clustering

Imports stringr, readr, WGCNA, dplyr, reshape2, data.table, patchwork, scales, igraph, flashClust, ggplot2, dcanr, cowplot, ggrepel, methods, SummarizedExperiment

Depends R (>= 4.3.0), ggalluvial

Config/testthat/edition 3

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

git_branch devel

git_last_commit b2881a4

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-01-17

Author Dario Tommasini [aut, cre] (ORCID:
<https://orcid.org/0000-0002-1214-6547>),
Brent Fogel [aut, ctb]

Maintainer Dario Tommasini <dtommasini0@gmail.com>

Contents

`multiWGCNA-package` *multiWGCNA: multiWGCNA*

Description

An R package for deeping mining gene co-expression networks in multi-trait expression data. Provides functions for analyzing, comparing, and visualizing WGCNA networks across conditions. multiWGCNA was designed to handle the common case where there are multiple biologically meaningful sample traits, such as disease vs wildtype across development or anatomical region.

Author(s)

Maintainer: Dario Tommasini <dtommasini0@gmail.com> (ORCID)

Authors:

- Brent Fogel [contributor]

`bidirectionalBestMatches` *Best matching modules*

Description

Find all the modules from dataset1 that have a best match to a module in dataset2 if that module in dataset2 is also a best match to the module in dataset1

Usage

```
bidirectionalBestMatches(comparisonList, plot = TRUE)
```

Arguments

`comparisonList`	a list with an elemnt "overlap", which is a data.frame resulting from a call to computeOverlapsFromWGCNA
`plot`	whether to generate a heatmap; default is TRUE

Value

A ggplot object

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
comparisonList = list()
comparisonList$overlaps = computeOverlapsFromWGCNA(astrocyte_networks$EAE, astrocyte_networks$WT)
bidirectionalBestMatches(comparisonList)
```

cleanDatExpr *cleanDatExpr*

Description

A function that converts a data.frame where row 1 is gene symbols to a numeric matrix where columns are genes and rows are samples for compatibility with most WGCNA functions.

Usage

```
cleanDatExpr(datExpr, checkGenesSamples = FALSE)
```

Arguments

datExpr	a data.frame were columns are samples and rows are samples and the gene symbols are in the first row
checkGenesSamples	call the WGCNA function checkGenesSamples?

Value

Returns a datExpr with rows as samples and columns as genes

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_se = eh_query[["EH8223"]]
datExpr = data.frame(X = rownames(assays(astrocyte_se)[[1]]), assays(astrocyte_se)[[1]])
cleanDatExpr(datExpr)
```

---

coexpressionLineGraph *Coexpression Line Graph*

Description

Plots a line graph showing the co-expression of selected genes across samples

Usage

```
coexpressionLineGraph(datExpr, splitBy = 1, fontSize = 2.15, colors = NULL)
```

Arguments

datExpr	a data.frame with genes as rows and samples as columns
splitBy	how much to split genes by on line graph
fontSize	the font size of the gene labels
colors	a vector of colors; default is random colors generated by colors function

Value

a ggplot object

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
datExpr = GetDatExpr(astrocyte_networks[[1]],
  genes = topNGenes(astrocyte_networks$EAE, "EAE_015", 20))
coexpressionLineGraph(datExpr) +
  geom_vline(xintercept = 20.5, linetype='dashed')
```

---

computeOverlapsFromWGCNA

computeOverlapsFromWGCNA

Description

Computes overlap between the modules of two objects of class WGCNA

Usage

```
computeOverlapsFromWGCNA(dataset1, dataset2)
```

Arguments

dataset1	an object of class WGCNA to compare with dataset2
dataset2	an object of class WGCNA to compare with dataset1

Value

Returns a data.frame showing the overlap results for modules from dataset1 with dataset2

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
computeOverlapsFromWGCNA(astrocyte_networks$EAE, astrocyte_networks$WT)
```

`constructNetworks` *constructNetworks: Construct all the weighted gene correlation networks*

Description

A high level function that returns all networks possible for a given experimental design

Usage

```
constructNetworks(
  datExpr,
  sampleTable,
  conditions1,
  conditions2,
  write = FALSE,
  alphaLevel = 0.05,
  plot = FALSE,
  ...
)
```

Arguments

`datExpr`	either a SummarizedExperiment object or data.frame with genes are rows and samples as columns
`sampleTable`	data.frame with sample names in first column and sample traits in the second and third column. First column should be called "Sample"
`conditions1`	first design conditions, ie healthy/disease
`conditions2`	second design conditions, ie frontal lobe/temporal lobe
`write`	write results out to files?
`alphaLevel`	significance value passed to findBestTrait function, default is 0.05
`plot`	plot modules? Default is false
`...`	Arguments to pass to blockwiseModules function

Value

A list of WGCNA objects, ie level one, two, and three networks.

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
autism_se = eh_query[["EH8219"]]
set.seed(1)
autism_se = autism_se[sample(rownames(autism_se), 500),]
sampleTable = colData(autism_se)
conditions1 = unique(sampleTable[,2])
conditions2 = unique(sampleTable[,3])
autism_networks = constructNetworks(autism_se, sampleTable, conditions1[[1]], conditions2[[1]],
  networkType = "signed", TOMType = "unsigned",
  power = 10, minModuleSize = 100, maxBlockSize = 25000,
  reassignThreshold = 0, minKMEtoStay = 0, mergeCutHeight = 0,
  numericLabels = TRUE, pamRespectsDendro = FALSE,
  deepSplit = 4, verbose = 3)
autism_networks[["combined"]]
```

---

diffCoexpression *Differential co-expresison analysis*

Description

Performs a differential co-expression ananlysis given an expression data.frame and a conditions vector

Usage

```
diffCoexpression(
  datExpr,
  conditions,
  geneList = NULL,
  plot = FALSE,
  method = c("pearson", "spearman"),
  removeFreeNodes = TRUE,
  labelSize = 0.5,
  labelDist = 0,
  shape = "circle",
  degreeForSize = FALSE,
```

```
  label = FALSE,
  onlyPositive = FALSE,
  z.threshold = NULL,
  FDR.threshold = 0.05,
  nodeSize = 3
)
```

Arguments

datExpr	a data.frame containing expression values
conditions	a vector containing conditions for the samples
geneList	vector of genes, will use all genes if NULL (default)
plot	plot a network?
method	either "pearson" or "spearman"
removeFreeNodes	remove free nodes from network?
labelSize	label size
labelDist	distance from labels to nodes
shape	shape of nodes
degreeForSize	should node size correspond to degree?
label	label nodes?
onlyPositive	only draw positive correlations?
z.threshold	z-score threshold
FDR.threshold	FDR threshold
nodeSize	size of node

Value

A list including a matrix of z-scores, a matrix of raw p-values, a matrix of adjusted p-values, and a summary data.frame

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_se = eh_query[["EH8223"]]
datExpr = assays(astrocyte_se)[[1]]
diffCoexpression(datExpr, c(rep(1,20), rep(2,16)),
  geneList = c("Gfap", "Vim", "Aspg", "Serpina3n", "Cp", "Osmr", "Cd44",
    "Cxcl10", "Hspb1", "Timp1", "S1pr3", "Steap4", "Lcn2"))
```

`diffModuleExpression` *Differential module expression*

Description

Runs (and plots) the differential module expression analysis

Usage

```
diffModuleExpression(
  WGCNAobject,
  geneList,
  design,
  plotTitle = NULL,
  mode = c("PC1", "Zscore"),
  testColumn = 2,
  refColumn = 3,
  test = c("ANOVA", "PERMANOVA"),
  plot = TRUE
)
```

Arguments

`WGCNAobject`	WGCNA object
`geneList`	vector of genes in WGCNAobject
`design`	the sampleTable
`plotTitle`	title for the plot
`mode`	either PC1 or Zscore, default is PC1
`testColumn`	the column of the sampleTable to be resolved
`refColumn`	the column of the sampleTable to be used as biological variation
`test`	statistical test to perform, either "ANOVA" or "PERMANOVA"
`plot`	generate a plot?

Value

a data.frame with the resulting p-values

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_se = eh_query[["EH8223"]]
sampleTable = colData(astrocyte_se)
astrocyte_networks = eh_query[["EH8222"]]
```

```
diffModuleExpression(astrocyte_networks[["combined"]],
  topNGenes(astrocyte_networks$combined, "combined_013"),
  sampleTable,
  test = "ANOVA",
  plotTitle = "combined_013",
  plot = TRUE)
```

`drawMultiWGCNAnetwork` *Draw multiWGCNA network*

Description

Draw a network where nodes are modules and edges represent significant gene overlap. Modules are sorted by levels 1, 2, and 3.

Usage

```
drawMultiWGCNAnetwork(
  WGCNAlist,
  comparisonList,
  moduleOfInterest,
  design,
  overlapCutoff = 0,
  padjCutoff = 1,
  removeOutliers = TRUE,
  alpha = 1e-50,
  layout = NULL,
  hjust = 0.4,
  vjust = 0.3,
  width = 0.5,
  colors = NULL
)
```

Arguments

`WGCNAlist`	list of WGCNA objects
`comparisonList`	the list of overlap comparisons ie from iterate(myNetworks, overlapComparisons, ...)
`moduleOfInterest`	module of interest, ie "combined_001"
`design`	the sampleTable design matrix
`overlapCutoff`	cutoff to remove module correspondences with less than this number of genes
`padjCutoff`	cutoff to remove module correspondences above this significance value
`removeOutliers`	remove outlier modules?
`alpha`	alpha level of significance

layout	layout of network to be passed to plot function of igraph object, defaults to multiWGCNA custom layout
hjust	horizontal justification of labels
vjust	vertical justification of labels
width	width of labels
colors	colors to use for modules, should be the same length as the number of WGCNA objects in the WGCNAlist. Defaults to random colors for each condition.

Value

an igraph plot

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_se = eh_query[["EH8223"]]
sampleTable = colData(astrocyte_se)
astrocyte_networks = eh_query[["EH8222"]]
results = list()
results$overlaps = iterate(astrocyte_networks, overlapComparisons, plot=FALSE)
drawMultiWGCNAnetwork(astrocyte_networks,
  results$overlaps,
  "combined_013",
  sampleTable)
```

GetDatExpr *Get expression data*

Description

Returns the expression data frame a WGCNA object as a data.frame

Usage

```
GetDatExpr(object, genes = NULL)
```

Arguments

object	An object of class WGCNA
genes	a list of genes to subset to; default is NULL

Value

a data.frame

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
datExpr = GetDatExpr(astrocyte_networks[[1]],
  genes = topNGenes(astrocyte_networks$EAE, "EAE_015", 20))
coexpressionLineGraph(datExpr) +
  geom_vline(xintercept = 20.5, linetype='dashed')
```

`getPreservation` *getPreservation*

Description

Performs a network preservation analysis

Usage

```
getPreservation(reference, test, nPermutations = 100, write = FALSE)
```

Arguments

`reference`	reference network of class WGCNA
`test`	test network of class WGCNA
`nPermutations`	number of permutations to perform; at least 50 permutations
`write`	write to file?

Value

a data.frame summarizing results of preservation analysis

Author(s)

Dario Tommasini

iterate *iterate: Iterate function across networks*

Description

A high level function that iterates functions across a list of WGCNA objects

Usage

```
iterate(WGCNAlist, FUN, ...)
```

Arguments

WGCNAlist	a vector of objects of type WGCNAobject
FUN	function to iterate, either overlapComparisons or preservationComparisons
...	argmuents to be passed on to overlapComparisons or preservationComparisons

Value

a comparison list from overlapComparisons or preservationComparisons

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
results = list()
iterate(astrocyte_networks, overlapComparisons, plot=FALSE)
```

makeTraitTable *Generate a trait table from a sample table*

Description

Generates a WGCNA-compatible trait table from a sampleTable dataframe

Usage

```
makeTraitTable(inputTable, column, detectNumbers = FALSE)
```

Arguments

inputTable	the sampleTable data.frame
column	the column from the sampleTable to use as traits
detectNumbers	whether to consider traits with numbers as numerical rather than categorical variables

Value

a data.frame with integer values denoting the categorical sample traits

Examples

```
sampleTable = data.frame(Sample = c(paste0("EAE", 1:10), paste0("WT", 1:10)),
                         Disease = c(rep("EAE", 10), rep("WT", 10)),
                         Region = c(rep(c("Cbl", "Sc"), 5)))
makeTraitTable(sampleTable, 2)
```

---

moduleComparisonPlot *Module comparison plot*

Description

A plotting function that returns a heatmap and barplot for a module

Usage

```
moduleComparisonPlot(overlapDf, dataset1, dataset2)
```

Arguments

overlapDf	a data.frame resulting from a call to computeOverlapsFromWGCNA
dataset1	an object of class WGCNA to compare with dataset2
dataset2	an object of class WGCNA to compare with dataset1

Value

Returns a ggplot object (flowplot and heatmap) showing the module correspondence between two objects of class WGCNA

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
overlapDf = computeOverlapsFromWGCNA(astrocyte_networks$EAE, astrocyte_networks$WT)
moduleComparisonPlot(overlapDf, astrocyte_networks$EAE, astrocyte_networks$WT)
```

---

moduleExpressionPlot *Plots an expression profile for a module*

Description

A plotting function that returns a heatmap and barplot for a module

Usage

```
moduleExpressionPlot(
  WGCNAobject,
  geneList,
  mode = c("PC1", "averageZscore"),
  legend = FALSE,
  title = NULL,
  clusterGenes = FALSE
)
```

Arguments

WGCNAobject	an object of class WGCNAobject
geneList	a vector of gene names to be extracted from WGCNAobject
mode	use first principal component or averageZscore?
legend	plot legend?
title	title of the plot
clusterGenes	cluster heatmap genes by hierarchical clustering?

Value

a patchworked ggplot object

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
moduleExpressionPlot(astrocyte_networks[["combined"]],
  geneList = topNGenes(astrocyte_networks$combined, "combined_013"))
```

`moduleToModuleHeatmap` *Module to module heatmap*

Description

Returns a heatmap where color corresponds to FDR-adjusted overlap (hypergeometric test) and the label corresponds to the number of overlapping genes

Usage

```
moduleToModuleHeatmap(
  comparisonDf,
  dataset1 = NULL,
  dataset2 = NULL,
  trait1 = NULL,
  trait2 = NULL,
  list1 = NULL,
  list2 = NULL,
  filterByTrait = FALSE,
  alphaLevel = 0.05
)
```

Arguments

`comparisonDf`	the data.frame output of computeOverlapFromWGCNA
`dataset1`	optional; WGCNA object for dataset 1
`dataset2`	optional; WGCNA object for dataset 2
`trait1`	optional; subset to modules correlated to this trait for dataset 1
`trait2`	optional; subset to modules correlated to this trait for dataset 2
`list1`	subset to this list of modules for dataset 1
`list2`	subset to this list of modules for dataset 2
`filterByTrait`	only plot for modules that correlate with some trait?
`alphaLevel`	the alpha level of significance for module-trait correlation, defaults to 0.05

Value

A ggplot object

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
overlapDf = computeOverlapsFromWGCNA(astrocyte_networks$EAE, astrocyte_networks$WT)
moduleToModuleHeatmap(overlapDf)
```

---

name *name: Name of WGCNAobject*

Description

Returns the name of a WGCNAobject.

Usage

```
name(WGCNAobject)
```

Arguments

WGCNAobject	an object of class WGCNA

Value

Returns the name of the WGCNA object, ie "EAE" for astrocyte_networks$EAE.

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
name(astrocyte_networks$EAE)
```

---

overlapComparisons *Overlap comparisons*

Description

Compares modules between two objects of type WGCNAobjects within a WGCNAobject list given the indices. Recommended to be used in conjunction with the iterate function.

Usage

```
overlapComparisons(
  comparisonList,
  WGCNAlist,
  first,
  second,
  element,
  plot = TRUE,
  write = FALSE
)
```

Arguments

comparisonList	a list passed by the iterate function
WGCNAlist	list of objects of class WGCNA
first	index of first WGCNA object
second	index of second WGCNA object
element	element position in the comparison list (passed by iterate function)
plot	generate plots?
write	write results to file?

Value

A list, in which the first element is a data.frame showing the overlap results and the second element is a data.frame showing the best matching modules between the two WGCNA objects.

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
results = list()
results$overlaps = iterate(astrocyte_networks, overlapComparisons, plot=FALSE)
```

performANOVA *Perform ANOVA*

Description

Test association between module expression to traits using ANOVA

Usage

```
performANOVA(datExpr, design, testCondition, refCondition, alphaLevel = 0.05)
```

Arguments

datExpr	expression data.frame
design	the sampleTable
testCondition	test column in sampleTable
refCondition	reference column in sampleTable
alphaLevel	the significance level

Value

a data.frame with p-values for each association

permutationTestResults *Permutation test results*

Description

The results of running the PreservationPermutationTest in the astrocyte vignette. This is provided since this function is quite slow. Please see the astrocyte vignette for more details.

Usage

```
data(permutationTestResults)
```

Format

A list of data.frames containing preservation results for each permutation

`preservationComparisonPlot` *Preservation Comparison Scatterplot*

Description

A plotting function that draws a scatterplot of preservation scores between two WGCNA objects

Usage

```
preservationComparisonPlot(
  preservationList,
  dataset1,
  dataset2,
  alphaLevel = 0.05,
  outliers = FALSE
)
```

Arguments

`preservationList`	a list resulting from a call to preservationComparisons
`dataset1`	an object of class WGCNAobject to compare with dataset2
`dataset2`	an object of class WGCNAobject to compare with dataset1
`alphaLevel`	alpha level of significance, default is 0.05
`outliers`	leave outlier modules? By default these are removed

Value

a ggplot object

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
results = list()
results$preservation=iterate(astrocyte_networks[c("EAE", "WT")],
  preservationComparisons,
  write=FALSE,
  plot=FALSE,
  nPermutations=2)
```

```
preservationComparisonPlot(results$preservation$EAE_vs_WT,
  astrocyte_networks$EAE,
  astrocyte_networks$WT)
```

`preservationComparisons` *Preservation comparisons*

Description

A high level function that performs a perservation comparison between two WGCNAobjects in a WGCNAlist, usually supplied by iterate function

Usage

```
preservationComparisons(
  comparisonList,
  WGCNAlist,
  first,
  second,
  element,
  plot = FALSE,
  write = FALSE,
  alphaLevel = 0.05,
  nPermutations = 100
)
```

Arguments

`comparisonList`	a list passed by the iterate function
`WGCNAlist`	list of objects of type WGCNAobject
`first`	index of first WGCNAobject
`second`	index of second WGCNAobject
`element`	element position in the comparison list (passed by iterate function)
`plot`	generate plots?
`write`	write results to file?
`alphaLevel`	alpha level of significance for module-trait correlation
`nPermutations`	number of permutations, defaults to 100

Value

a list of preservation comparisons results across levels 1, 2, 3

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
results = list()
iterate(astrocyte_networks[c("EAE", "WT")],
  preservationComparisons,
  write=FALSE,
  plot=FALSE,
  nPermutations=2)
```

`PreservationPermutationTest`
PreservationPermutationTest

Description

Performs a permutation test to determine if a null distribution of expected preservation scores for modules in this dataset if the labels were randomly assigned. Please look at the astrocyte vignette for more info.

Usage

```
PreservationPermutationTest(
  referenceDatExpr,
  design,
  constructNetworksIn,
  testPreservationIn,
  nPermutations = 100,
  nPresPermutations = 100,
  ...
)
```

Arguments

`referenceDatExpr`	the combined datExpr
`design`	the sampleTable
`constructNetworksIn`	the condition to use for network construction, e.g. for the astrocyte data, this is "EAE"

testPreservationIn
: the condition to use for testing preservation, e.g. for the astrocyte data, this was "WT"

nPermutations
: the number of permutations to perform for permutation test

nPresPermutations
: the number of permutations to perform in modulePreservation function

...
: arguments to pass to blockwiseModules function for network construction (should be the same as used for constructing the original network)

Value

A list of data.frames with preservation results for each permutation

Author(s)

Dario Tommasini

Examples

```
## Not run:
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
astrocyte_se = eh_query[["EH8223"]]
sampleTable = colData(astrocyte_se)
results = list()
results$permutation.test = PreservationPermutationTest(astrocyte_networks$combined@datExpr[sample(17000,3000),]
                                                       sampleTable,
                                       constructNetworksIn = "EAE", # Construct networks using EAE samples
                                       testPreservationIn = "WT", # Test preservation of disease samples in WT sampl
                                       nPermutations = 10, # Number of permutations for permutation test
                                       nPresPermutations = 10, # Number of permutations for modulePreservation funct
                                              networkType = "signed", TOMType = "unsigned",
                                       power = 12, minModuleSize = 100, maxBlockSize = 25000,
                                       reassignThreshold = 0, minKMEtoStay = 0, mergeCutHeight = 0,
                                            numericLabels = TRUE, pamRespectsDendro = FALSE,
                                                       deepSplit = 4, verbose = 3
                                                       )

## End(Not run)
```

PreservationScoreDistribution

PreservationScoreDistribution

Description

Extracts the preservation score distribution from the results of PreservationPermutationTest.

Usage

```
PreservationScoreDistribution(preservationData, moduleOfInterestSize)
```

Arguments

preservationData	the results from PreservationPermutationTest
moduleOfInterestSize	the number of genes in your module of interest

Value

A data.frame with Z-summary preservation scores of the module from each permutation and the corresponding module size

Author(s)

Dario Tommasini

Examples

```
# Remove outlier modules
permutationTestResultsFiltered = lapply(permutationTestResults, function(x) x[!x$is.outlier.module,])

# Find preservation score distribution for a given module size
scores.summary = PreservationScoreDistribution(permutationTestResultsFiltered,
                                               moduleOfInterestSize = 303)
```

---

runDME *Run differential module expression*

Description

A wrapper to run diffModuleExpression on all the modules in a network

Usage

```
runDME(
  WGCNAobject,
  design,
  alphaLevel = 0.05,
  testCondition = NULL,
  refCondition = NULL,
  p.adjust = "fdr",
  plot = FALSE,
  test = c("ANOVA", "PERMANOVA"),
  write = FALSE,
  out = NULL
)
```

Arguments

WGCNAobject	object of class WGCNA with the modules to run DME on
design	the sampleTable
alphaLevel	level of significance
testCondition	the column of the sampleTable to be resolved
refCondition	the column of the sampleTable to be used as biological variation
p.adjust	adjust for multiple comparisons, argument to pass to p.adjust function
plot	generate a plot?
test	statistical test to perform, either "ANOVA" or "PERMANOVA"
write	write results to a file?
out	file name for DME plots, only used if write is TRUE

Value

a data.frame summarizing the results of the analysis

Author(s)

Dario Tommasini

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_se = eh_query[["EH8223"]]
sampleTable = colData(astrocyte_se)
astrocyte_networks = eh_query[["EH8222"]]
runDME(astrocyte_networks[["combined"]],
  design = sampleTable,
  p.adjust = "fdr",
  refCondition = "Region",
  testCondition = "Disease")
```

---

summarizeResults *summarizeResults: Summarize results from a results list object*

Description

Prints (or writes) a summary of the results from a results list object

Usage

```
summarizeResults(
  myNetworks,
  results,
  alphaLevel = 0.05,
  write = FALSE,
  outputFile = "results.txt"
)
```

Arguments

myNetworks	a list of WGCNAobjects
results	results list
alphaLevel	alpha level of significance
write	write to file?
outputFile	name of output file, defaults to results.txt

Value

prints a summary of results from the multiWGCNA analysis

TOMFlowPlot *TOMFlowPlot*

Description

Plots a sankey flow diagram showing the movement of genes from one WGCNA to another WGCNA. Uses the ggalluvial framework.

Usage

```
TOMFlowPlot(
  WGCNAlist,
  networks,
  toms,
  genes_to_label,
  alpha = 0.1,
  color = "black",
  width = 0.05
)
```

Arguments

WGCNAlist	list of WGCNA objects
networks	list of network names of length 2
toms	a list of TOM distance objects of length 2
genes_to_label	genes to label across two networks
alpha	alpha of flows
color	color of flows
width	width of the strata

Value

a ggplot object

Author(s)

Dario Tommasini

---

topNGenes *topNGenes: Top N genes of a module*

Description

Returns the top N number of genes of a module. All genes returned if no number is specified. Genes are in order of intramodular connectivity.

Usage

```
topNGenes(WGCNAobject, module, nGenes = NULL)
```

Arguments

WGCNAobject	an object of class WGCNA
module	the name of the module in WGCNAobject
nGenes	an integer from 1 to module size; returns all genes if left NULL

Value

a character vector of the genes/probes in the module

Examples

```
library(ExperimentHub)
eh = ExperimentHub()
eh_query = query(eh, c("multiWGCNAdata"))
astrocyte_networks = eh_query[["EH8222"]]
topNGenes(astrocyte_networks$EAE, "EAE_015", nGenes = 10)
```

WGCNA-class *The WGCNA Class*

Description

The WGCNA class is the main class used in multiWGCNA to store results from a weighted gene co-expression nework analysis. These include the original unaltered expression data used as input, connectivity metrics, module assignment, input sample conditions, trait

Value

NA

Slots

`datExpr` The expression data, connectivity data, and module assignment

`conditions` A data.frame with integer conditions for WGCNA

`trait` A data.frame showing pearson correlation values to traits

`moduleEigengenes` A data.frame of module eigengenes for each module across samples

`outlierModules` A vector of modules classified by our algorithm as being driven by sample outliers

Index