Package ‘sigaR’

March 23, 2017

Type Package
Title Statistics for Integrative Genomics Analyses in R
Version 1.22.0
Date 2016-10-04
Author Wessel N. van Wieringen <w.vanwieringen@vumc.nl>
Maintainer Wessel N. van Wieringen <w.vanwieringen@vumc.nl>
Description Facilitates the joint analysis of high-throughput data from multiple molecular levels. Contains functions for manipulation of objects, various analysis types, and some visualization.
License GPL (>= 2)
LazyLoad yes
URL http://www.few.vu.nl/~wvanwie
Depends Biobase, CGHbase, methods, mvtwonorm,
Imports corpcor (>= 1.6.2), graphics, igraph, limma, marray, MASS,
penalized, quadprog, snowfall, stats
Suggests CGHcall
biocViews Microarray, DifferentialExpression, aCGH, GeneExpression,
Pathways
NeedsCompilation no

R topics documented:
sigaR-package ......................................................... 2
  .RCMloss-method ............................................. 3
cghCall2cghSeg .................................................... 4
cghCall2maximumSubset ........................................ 5
cghCall2order ..................................................... 6
cghCall2subset ................................................... 7
cghCall2weightedSubset ........................................ 8
cghSeg2order ....................................................... 9
cghSeg2subset ..................................................... 10
cghSeg2weightedSubset ......................................... 11
cisEffectPlot ....................................................... 12
cisEffectTable ..................................................... 13
cisEffectTest ...................................................... 14
sigaR-package

Description

The package facilitates several types of integrative analysis of high-throughput data from various molecular levels. In addition, it includes functions for data management and visualization.
Details

Package: sigaR
Type: Package
Version: 1.0
Date: 2011-04-15
License: What license is it under?
LazyLoad: yes

Author(s)

Author: Wessel N. van Wieringen
Maintainer: Wessel N. van Wieringen <w.vanwieringen@vumc.nl>

References


Description

Internal function.

Note

Not to be called by the user.
cghCall2cghSeg

Genomic ordering of cghSeg-objects.

Description

Transforms a cghCall-object to a cghSeg-object, by removing the slots present in the former but not in the latter.

Usage

cghCall2cghSeg(CNdata, verbose=TRUE)

Arguments

CNdata Object of class cghCall.
verbose Logical indicator: should intermediate output be printed on the screen?

Value

Object of class cghSeg.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

cghCall, cghSeg.

Examples

# load data
data(pollackCN16)

# reduce the cghCall-object to a cghSeg-object
pollackCN16seg <- cghCall2cghSeg(pollackCN16)
cghCall2maximumSubset  Maximum subsetting cghCall-objects.

Description

Limit an cghCall object to a subset of its features, selecting those features with the most deviating copy number signal.

Usage

cghCall2maximumSubset(CNdata, featuresAndWeights, chr, bpstart, bpend, ncpus = 1, verbose=TRUE)

Arguments

CNdata Object of class cghCall.
featuresAndWeights Object of class list. Each list item is a matrix. The first column of this matrix contains the row numbers of features to be maintained in the cghCall-object. The second column contains the weights of each features, to be used in the calculation of the weighted average copy number signal.
chr Column in the slot featureData of the cghCall-object specifying the chromosome information of the features.
bpstart Column in the slot featureData of the cghCall-object specifying the start basepair information of the features.
bpend Column in the slot featureData of the cghCall-object specifying the end basepair information of the features.
cpus Number of cpus to be used in computations.
verbose Logical indicator: should intermediate output be printed on the screen?

Details

Per entry of the featuresAndWeights-object and per sample the feature with the maximum absolute segmented DNA copy number signal is selected.

Value

Object of class cghCall, restricted to the specified subset of features.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References

cghCall2order

Genomic ordering of cghCall-objects.

Description

Orders the features within a cghCall-object in accordance with their genomic order.

Usage

cghCall2order(CNdata, chr, bpstart, verbose=TRUE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNdata</td>
<td>Object of class cghCall.</td>
</tr>
<tr>
<td>chr</td>
<td>Column in the slot featureData of the cghCall-object specifying the chromosome information of the features.</td>
</tr>
<tr>
<td>bpstart</td>
<td>Column in the slot featureData of the cghCall-object specifying the start basepair information of the features.</td>
</tr>
<tr>
<td>verbose</td>
<td>Logical indicator: should intermediate output be printed on the screen?</td>
</tr>
</tbody>
</table>

Value

Object of class cghCall, now genomically ordered.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


Examples

# load data
data(pollackCN16)

# extract genomic information from ExpressionSet-object
chr <- fData(pollackCN16)[,1]
bpstart <- fData(pollackCN16)[,2]
bpend <- fData(pollackCN16)[,3]

# find unique genomic locations
uniqInfo <- uniqGenomicInfo(chr, bpstart, bpend, verbose = FALSE)

# subset cghCall-object to features with unique genomic locations
pollackCN16 <- cghCall2maximumSubset(pollackCN16, uniqInfo, 1, 2, 3)
cghCall2subset

See Also

cghCall.

Examples

# load data
data(pollackCN16)

# order the copy number data genomically
cghCall2order(pollackCN16, 1, 2)

cghCall2subset

Subsetting cghCall-objects.

Description

Limit an cghCall object to a subset of its features.

Usage

cghCall2subset(CNdata, featureSubset, verbose=TRUE)

Arguments

CNdata Object of class cghCall.
featureSubset Object of class numeric, containing the row numbers of features to be main-
tained in the cghCall-object.
verbose Logical indicator: should intermediate output be printed on the screen?

Value

Object of class cghCall, restricted to the specified subset of features.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also
cghCall.

Examples

# load data
data(pollackCN16)

# order the copy number data genomically
cghCall2order(pollackCN16, 1, 2)
cghCall2weightedSubset

Weighted subsetting cghCall-objects.

Description
Limit an cghCall object to a subset of its features, using weighted averaging of the copy number signal.

Usage
cghCall2weightedSubset(CNdata, featuresAndWeights, chr, bpstart, bpend, ncpus = 1, verbose=TRUE)

Arguments
- **CNdata**: Object of class cghCall.
- **featuresAndWeights**: Object of class list. Each list item is a matrix. The first column of this matrix contains the row numbers of features to be maintained in the cghCall-object. The second column contains the weights of each features, to be used in the calculation of the weighted average copy number signal.
- **chr**: Column in the slot featureData of the cghCall-object specifying the chromosome information of the features.
- **bpstart**: Column in the slot featureData of the cghCall-object specifying the start basepair information of the features.
- **bpend**: Column in the slot featureData of the cghCall-object specifying the end basepair information of the features.
- **ncpus**: Number of cpus to be used in computations.
- **verbose**: Logical indicator: should intermediate output be printed on the screen?

Value
Object of class cghCall, restricted to the specified subset of features.

Warning
The phenoData, experimentData, and other slots of the cghCall-object are currently not passed on to the subsetted object.

Note
This is a more intricate version of the cghCall2subset function. They exists parallel because this function is (much) slower than its counterpart.

Author(s)
Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>
cghSeg2order

Genomic ordering of cghSeg-objects.

Description

Orders the features within a cghSeg-object in accordance with their genomic order.

Usage

cghSeg2order(CNdata, chr, bpstart, verbose=TRUE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNdata</td>
<td>Object of class cghSeg.</td>
</tr>
<tr>
<td>chr</td>
<td>Column in the slot featureData of the cghSeg-object specifying the chromosome information of the features.</td>
</tr>
<tr>
<td>bpstart</td>
<td>Column in the slot featureData of the cghSeg-object specifying the start base-pair information of the features.</td>
</tr>
<tr>
<td>verbose</td>
<td>Logical indicator: should intermediate output be printed on the screen?</td>
</tr>
</tbody>
</table>

Value

Object of class cghSeg, now genomically ordered.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>
References


See Also
cghSeg.

Examples

# load data
data(pollackCN16)

# transform the cghCall-object to a cghSeg-object
pollackCN16 <- cghCall2cghSeg(pollackCN16)

# order the copy number data genomically
pollackCN16 <- cghSeg2order(pollackCN16, 1, 2)

cghSeg2subset

Subsetting cghSeg-objects.

Description

Limit an cghSeg object to a subset of its features.

Usage

cghSeg2subset(CNdata, featureSubset, verbose=TRUE)

Arguments

CNdata Object of class cghSeg.
featureSubset Object of class numeric, containing the row numbers of features to be maintained in the cghSeg-object.
verbose Logical indicator: should intermediate output be printed on the screen?

Value

Object of class cghSeg, restricted to the specified subset of features.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References

See Also

cghSeg.

Examples

# load data
data(pollackCN16)

# transform the cghCall-object to a cghSeg-object
pollackCN16 <- cghCall2cghSeg(pollackCN16)

# subset the copy number data
pollackCN16 <- cghSeg2subset(pollackCN16, c(1:50))

cghSeg2weightedSubset  Weighted subsetting cghSeg-objects.

Description

Limit an cghSeg object to a subset of its features, using weighted averaging of the copy number signal.

Usage

cghSeg2weightedSubset(CNdata, featuresAndWeights, chr, bpstart, bpend, ncpus = 1, verbose=TRUE)

Arguments

CNdata  Object of class cghSeg.
featuresAndWeights
        Object of class list. Each list item is a matrix. The first column of this matrix contains the row numbers of features to be maintained in the cghSeg-object. The second column contains the weights of each features, to be used in the calculation of the weighted average copy number signal.
chr     Column in the slot featureData of the cghSeg-object specifying the chromosome information of the features.
bpstart Column in the slot featureData of the cghSeg-object specifying the start base-pair information of the features.
bpend   Column in the slot featureData of the cghSeg-object specifying the end base-pair information of the features.
cpus    Number of cpus to be used in computations.
verbose Logical indicator: should intermediate output be printed on the screen?

Value

Object of class cghSeg, restricted to the specified subset of features.
cisEffectPlot

Warning

The phenoData, experimentData, and other slots of the cghSeg-object are currently not passed on to the subsetted object.

Note

This is a more intricate version of the cghSeg2subset function. They exist parallel because this function is (much) slower than its counterpart.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

cghSeg2subset

Examples

# load data
data(pollackCN16)

# extract genomic information from ExpressionSet-object
chr <- fData(pollackCN16)[,1]
bpstart <- fData(pollackCN16)[,2]
bpend <- fData(pollackCN16)[,3]

# find unique genomic locations
uniqInfo <- uniqGenomicInfo(chr, bpstart, bpend, verbose = FALSE)

# transform the cghCall-object to a cghSeg-object
pollackCN16 <- cghCall2cghSeg(pollackCN16)

# subset cghSeg-object to features with unique genomic locations
pollackCN16 <- cghSeg2weightedSubset(pollackCN16, uniqInfo, 1, 2, 3)

cisEffectPlot DNA-mRNA plot

Description

A variant on the boxplot, plotting the gene expression against the DNA copy number data. For each individual an open blue circle per call is plotted, all with their centerpoint at the height of the individual’s expression level. The radius of the circles is proportional to the corresponding call probabilities. Call probabilities equal to zero reduce circles to dots. The red filled circles have a radius proportional to the estimated expected call probabilities, with their centerpoints at the estimated mean expression for the respective call.
Usage

cisEffectPlot(geneId, CNdata, GEdata, verbose=FALSE)

Arguments

geneId  Indicator of the gene to be plotted. Indicator refers to the row in the ExpressionSet-object.

CNdata  Object of class cghCall, containing (among others) annotation and call probabilities. Features should be matched with those of the accompanying ExpressionSet-object (as may be done using the matchAnn2Ann-function).

GEdata  Object of class ExpressionSet. Features should be matched with those of the accompanying cghCall-object (as may be done using the matchAnn2Ann-function).

verbose  Logical indicator: should intermediate output be printed on the screen?

Note

This function is a rewritten version of the intCNGEan.plot function of the intCNGEan-package.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

boxplot, cisEffectTune, cisEffectTest, matchAnn2Ann

Examples

# load data
data(pollackCN16)
data(pollackGE16)

# plot DNA copy number vs. gene expression.
cisEffectPlot(225, pollackCN16, pollackGE16)

cisEffectTable  Table of cis-effect test results

Description

Function to display the results of cisEffectTest-function in a table-format. Table may be restricted to a specified number of genes and sorted by relevant statistics.

Usage

cisEffectTable(testRes, number=10, sort.by=NULL)
cisEffectTest

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>testRes</td>
<td>Object of class cisTest as produced by the cisEffectTest-function.</td>
</tr>
<tr>
<td>number</td>
<td>Number of genes whose results are to be included in the table.</td>
</tr>
<tr>
<td>sort.by</td>
<td>Character indicating how the table is to be sorted: NULL no sorting (genomic order), p.value, R2 or effect sort the table by the corresponding statistic.</td>
</tr>
</tbody>
</table>

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

cisEffectTest

Examples

```r
# load data
data(pollackCN16)
data(pollackGE16)

# test cis-effect of DNA copy number on gene expression levels
cisRes <- cisEffectTest(pollackCN16, pollackGE16, 1:nrow(pollackGE16), 1, nPerm=25)

# display top results
cisEffectTable(cisRes, number=10, sort.by="R2")
```

cisEffectTest  Nonparametric testing for copy number induced differential gene expression.

Description

A nonparametric test for the detection of copy number induced differential gene expression. The test incorporates the uncertainty of the calling of genomic aberrations: weighted version of well-known test statistics are used. An efficient permutation re-sampling procedure is used for p-value calculation. The test statistics may be "shrunken" to borrow information across neighboring genes that share the same copy number signature.

Usage

```r
cisEffectTest(CNdata, GEdata, genes2test=NULL, GEchr, analysisType="univariate", testStatistic="wcvm", nPerm = 10000, lowCiThres = 0.1, verbose=TRUE)
```
cisEffectTest 15

Arguments

**Cndata**
Object of class cghCall, containing (among others) annotation and call probabilities. Features should be matched with those of the accompanying ExpressionSet-object (as may be done using the matchAnn2Ann-function).

**GEdata**
Object of class ExpressionSet. Features should be matched with those of the accompanying cghCall-object (as may be done using the matchAnn2Ann-function).

**genes2test**
Numeric indicator vector containing row number of genes for which the DNA copy number cis-effect should be tested. The function cisEffectTune yields an optimal selection.

**GEchr**
Column in the slot featureData of the ExpressionSet-object GEdata specifying the chromosome information of the features.

**analysisType**
Indicator to determine whether the test statistic should be "shrunken" within a region. Either "univariate" (no shrinkage) or "regional" (shrinkage).

**testStatistic**
Test statistic to be used, either "wcvm" or "wmw", the weighted Cramer-Von Mises and the weighted Mann-Whitney test statistic, respectively.

**nPerm**
Number of permutations used for the p-value calculation.

**lowCiThres**
A value between 0 and 1. Determines speed of efficient p-value calculation. Genes with a probability smaller than 0.001 of a p-value smaller than eff.p.val.thres are discarded from the permutation analysis and their p-value is set equal to 1. Should be chosen in accordance with the FDR-threshold for significance.

**verbose**
Logical indicator: should intermediate output be printed on the screen?

Value
Object of class cisTest.

Note
This function is a rewritten version of the intCNGEan.test function of the intCNGEan-package.

Author(s)
Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References

See Also
matchAnn2Ann, cisEffectTune, cisEffectTable, cisEffectPlot

Examples

```r
# load data
data(pollackCN16)
data(pollackGE16)

# test cis-effect of DNA copy number on gene expression levels
cisRes <- cisEffectTest(pollackCN16, pollackGE16, 1:nrow(pollackGE16), 1, nPerm=25)
```
cisEffectTune

Pre-test and tuning.

Description

Decides which test to perform: loss vs. no-loss (tumor supressor) or no-gain vs gain (proto-onco). Followed by a tuning algorithm that enhances the overall power of the FDR procedure by excluding genes with either unbalanced (many samples having a high call probability of, say, a loss) or imprecise (many call probabilities close to 0.5) soft calls, which is likely to increase the probability of detection for genes with a more favorable call probability distribution.

Usage

cisEffectTune(CNdata, GEdata, testStatistic, nGenes=250, nPerm=250, minCallProbMass=0.10, verbose=TRUE)

Arguments

- **CNdata**: Object of class `cghCall`, containing (among others) annotation and call probabilities. Features should be matched with those of the accompanying `ExpressionSet` object (as may be done using the `matchAnn2Ann` function).
- **GEdata**: Object of class `ExpressionSet`. Features should be matched with those of the accompanying `cghCall` object (as may be done using the `matchAnn2Ann` function).
- **testStatistic**: Test statistic to be used, either "wcvm" or "wmw", the weighted Cramer-Von Mises and the weighted Mann-Whitney test statistic, respectively.
- **nGenes**: Number of genes used for tuning.
- **nPerm**: Number of permutation used for tuning.
- **minCallProbMass**: A number inbetween 0 and 1. Genes with a marginal call probabilities in one of the classes smaller than `minCallProbMass` are discarded from further analysis. Effectively, this ensures identifiability of copy number effect on expression.
- **verbose**: Boolean to suppress output, either `FALSE` and `TRUE`.

Value

A numeric-object with the genes selected for testing. Numbering corresponds to genes of the pre-tuned, but matched data set.

Note

This function is a rewritten version of the `intCNGEan.tune` function of the `intCNGEan-package`.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References

cisTest-class

See Also

matchAnn2Ann, cisEffectTest

Examples

# load data
data(pollackCN16)
data(pollackGE16)

# select genes that are likely to have a significant genomic cis-effect on expression levels
genes2test <- cisEffectTune(pollackCN16, pollackGE16, "wmw", nGenes=50, nPerm=50)

cisTest-class

Class "cisTest" for storing the results of the function cisEffectTest.

Description

The class cisTest is the output of a call to cisEffectTest. It stores results from a hypothesis test.

Slots

geneInfo: Object of class "data.frame". E.g., annotation information of genes.
geneId: Object of class "numeric". Row number in ExpressionSet-object used in {cisEffectTest}, corresponding to a gene.
comparison: Object of class "numeric". Indicator of test performed, either "1" (loss vs. no-loss) or "2" (no-gain vs. gain).
av.prob1: Object of class "numeric". The estimated marginal call probability.
av.prob2: Object of class "numeric". The estimated marginal call probability.
effectSize: Object of class "numeric". Estimated genomic cis-effect on gene expression.
R2: Object of class "numeric". Percentage of explained variation in expression levels by .
regId: Object of class "numeric". Indicator for the region (NULL in the regional-analysis).
beginReg: Object of class "numeric". Row number in ExpressionSet-object corresponding to the first gene of the region (NULL in the regional-analysis).
endReg: Object of class "numeric". Row number in ExpressionSet-object corresponding to the last gene of the region (NULL in the regional-analysis).
shrinkage: Object of class "numeric". Amount of shrinkage applied in the regional analysis (NULL in the regional-analysis).
p.value: Object of class "numeric". P-value for the non-parametric test of the genomic cis-effect on expression levels.
adjP.value: Object of class "numeric". BH-multiple testing correct p-values.

analysisType: Indicator whether the test statistic has been "shrunken" within a region. Either "univariate" (no shrinkage) or "regional" (shrinkage).

testStatistic: Test statistic used, either "wcvm" or "wmw", the weighted Cramer-Von Mises and the weighted Mann-Whitney test statistic, respectively.
nPerm: Number of permutations used for the p-value calculation.
CNGEheatmaps

Methods

cisEffectTable  signature(object = "cisTest"): Prints the test results.

Author(s)

Wessel van Wieringen: <w.vanwieringen@vumc.nl>

See Also

cisEffectTest

Examples

showClass("cisTest")

CNGEheatmaps  Parallel CN and GE heatmap plotting

Description

Heatmaps of DNA copy number and gene expression data are plotted together.

Usage

CNGEheatmaps(CNdata, GEdata, location = "mode", colorbreaks = "equiquantiles")

Arguments

CNdata  Object of class cghCall, containing (among others) annotation and call probabilities. Features should be matched with those of the accompanying ExpressionSet-object (as may be done using the matchCGHcall2ExpressionSet-function).

GEdata  Object of class ExpressionSet. Features should be matched with those of the accompanying cghCall-object (as may be done using the matchCGHcall2ExpressionSet-function).

location  Parameter (median, mean, or mode) specifying how the center of the gene expression heatmap color-scheme is determined.

colorbreaks  Parameter specifying how the color distribution of the gene expression heatmap is determined, either equiquantiles or equidistant.

Details

The DNA copy number data heatmap is generated as follows. The DNA copy number data are used to determine the genomic segments exhibiting no difference in DNA copy number between the array elements that map to that segment. This resembles the dimension reduction technique employed in the CGHregions-package. Consequently, within a segment the DNA copy number for one sample is constant, but may vary between samples. Note that a region may comprise of a whole chromosome, but also of a focal amplification. It is the DNA copy number signature of the segments that is depicted in the heatmap of the DNA copy number data.

For the gene expression heatmap segments as constructed for the array CGH data are adopted. For each segment-sample combination the expression levels of the genes that map to that segment are averaged. Next, the gene expression data is also collapsed to the segment format. It is this collapsed and averaged expression data that is depicted in the corresponding heatmap.
entropyTest

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

cghCall, ExpressionSet, matchCGHcall2ExpressionSet, profilesPlot.

Examples

# load data
data(pollackCN16)
data(pollackGE16)

# plot heatmaps
CNGEheatmaps(pollackCN16, pollackGE16, location = "mode", colorbreaks = "equiquantiles")


test

One-sided two-sample test for entropy comparison

Description

A one-sided two-sample test compares the entropy of a (high-dimensional) multivariate random variable between two groups. The test is one-sided: one group is a priori suspected to have a larger entropy. The null distribution is obtained via an efficient permutation resampling algorithm.

Usage

entropyTest(Y, id, nPerm = 1000, method = "normal", k0 = 1, k1 = 1, center = TRUE, lowCiThres=0.10, ncpus=1, verbose=FALSE)

Arguments

Y (High-dimensional) matrix. Rows are assumed to represent the samples, and columns represent the samples' genes or traits.

id An indicator variable for the two groups to be compared. The groups should be coded as 0 and 1. There is an asymmetric interest in the groups: the group indicated by 1 is believed to exhibit a larger entropy.

nPerm Number of permutations.

method Distributional assumption under which entropy is to be estimated.

k0 k-nearest neighbor parameter for group comprising of samples indicated by a zero in the indicator variable id.

k1 k-nearest neighbor parameter for group comprising of samples indicated by a one in the indicator variable id.
center Logical indicator: should the columns of Y be centered around zero?
lowCiThres A value between 0 and 1. Determines speed of efficient p-value calculation. If the probability of a p-value being below lowCiThres is smaller than 0.001 (read: the test is unlikely to become significant), the permutation analysis is terminated and a p-value of 1.00 is reported.
ncpus Number of cpus used for the permutations.
verbose Logical indicator: should intermediate output be printed on the screen?

Value
Object of entTest-class.

Author(s)
Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References

See Also
hdEntropy

Examples
# load data
data(pollackGE16)
Y <- exprs(pollackGE16)

# assign samples to groups
id <- sample(c(0,1), 41, replace=TRUE)

# perform testing and print test results
testRes <- entropyTest(t(Y), id, nPerm = 5, method="knn")
summary(testRes)

entTest-class Class "entTest" for storing the results of the function entropyTest.

Description
The class entTest is the output of a call to entropyTest. It stores results from a hypothesis test.
SLOTS

statistic: Object of class "numeric". Observed test statistic (i.e., estimated mutual information).
p.value: Object of class "numeric". P-value for the mutual information test.
null.dist: Object of class "numeric". The permutation null distribution for the test statistic.
nperm: Object of class "numeric". Number of permutation used for p-value calculation.
remark: Object of class "character". Tells whether the permutation algorithm was terminated prematurely or not.

METHODS

summary signature(object = "entTest"): Prints the test results.

Author(s)

Wessel van Wieringen: <w.vanwieringen@vumc.nl>

See Also

entTest

Examples

showClass("entTest")

expandMatching2singleIDs

Expand matching to single entries

Description

In case a feature of platform 1 has been matched to multiple features of another platform, instead of averaging the data from these features, one may consider maintaining all features, each matched individually the feature of platform 1. This function modifies the results from the matching function matchAnn2Ann to facilitate this. The result can than directly be used in the subsetting functions cghCall2weightedSubset and ExpressionSet2weightedSubset.

Usage

expandMatching2singleIDs(matchedIDs)

Arguments

matchedIDs An object of class list, as returned by the matchAnn2Ann-function.

Value

An object of class list, similar to that returned by the matchAnn2Ann-function.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>
References


See Also

`matchAnn2Ann`, `cghCall2weightedSubset`, `ExpressionSet2weightedSubset`.

Examples

```r
# load data
data(pollackCN16)
data(pollackGE16)

# extract genomic information from cghCall-object
chr1 <- fData(pollackCN16)[,1]
bpstart1 <- fData(pollackCN16)[,2]
bpend1 <- fData(pollackCN16)[,3]

# extract genomic information from ExpressionSet-object
chr2 <- fData(pollackGE16)[,1]
bpstart2 <- fData(pollackGE16)[,2]
bpend2 <- fData(pollackGE16)[,3]

# match features from both platforms
matchedFeatures <- matchAnn2Ann(chr1, bpstart1, bpend1, chr2, 
bpstart2, bpend2, method = "distance", maxDist = 10000)

# expand
matchedFeatures <- expandMatching2singleIDs(matchedFeatures)
```

---

**ExpressionSet2order**

**Genomic ordering of ExpressionSet-objects.**

**Description**

Orders the features within an *ExpressionSet*-object in accordance with their genomic order.

**Usage**

`ExpressionSet2order(GEdata, chr, bpstart, verbose=TRUE)`

**Arguments**

- **GEdata**: Object of class *ExpressionSet*.
- **chr**: Column in the slot featureData of the *ExpressionSet*-object specifying the chromosome information of the features.
- **bpstart**: Column in the slot featureData of the *ExpressionSet*-object specifying the start basepair information of the features.
- **verbose**: Logical indicator: should intermediate output be printed on the screen?
Value

Object of class `ExpressionSet`, now genomically ordered.

Author(s)

Wessel N. van Wieringen: `<w.vanwieringen@vumc.nl>`

See Also

`ExpressionSet`

Examples

```r
# load data
data(pollackGE16)

# order the copy number data genomically
pollackGE16 <- ExpressionSet2order(pollackGE16, 1, 2)
```

---

**ExpressionSet2subset**  
*Subsetting ExpressionSet-objects.*

Description

Limit an `ExpressionSet` object to a subset of its features.

Usage

```r
ExpressionSet2subset(GEdata, featureSubset, verbose=TRUE)
```

Arguments

- **GEdata**: Object of class `ExpressionSet`.
- **featureSubset**: Object of class numeric, containing the row numbers of features to be maintained in the `ExpressionSet`-object.
- **verbose**: Logical indicator: should intermediate output be printed on the screen?

Value

Object of class `ExpressionSet`, restricted to the specified subset of features.

Author(s)

Wessel N. van Wieringen: `<w.vanwieringen@vumc.nl>`

See Also

`ExpressionSet`
### Examples

```r
# load data
data(pollackGE16)

# order the copy number data genomically
pollackGE16 <- ExpressionSet2subset(pollackGE16, c(1:50))
```

---

### ExpressionSet2weightedSubset

**Weighted subsetting ExpressionSet-objects.**

#### Description

Limit an `ExpressionSet` object to a subset of its features, using weighted averaging of the expression signal.

#### Usage

```r
ExpressionSet2weightedSubset(GEdata, featuresAndWeights, chr, bpstart, bpend, ncpus = 1, verbose=TRUE)
```

#### Arguments

- `GEdata` Object of class `ExpressionSet`.
- `featuresAndWeights` Object of class `list`. Each list item is a `matrix`. The first column of this `matrix` contains the row numbers of features to be maintained in the `ExpressionSet`-object. The second column contains the weights of each feature, to be used in the calculation of the weighted average gene expression signal.
- `chr` Column in the slot `featureData` of the `ExpressionSet`-object specifying the chromosome information of the features.
- `bpstart` Column in the slot `featureData` of the `ExpressionSet`-object specifying the start basepair information of the features.
- `bpend` Column in the slot `featureData` of the `ExpressionSet`-object specifying the end basepair information of the features.
- `ncpus` Number of cpus to be used in computations.
- `verbose` Logical indicator: should intermediate output be printed on the screen?

#### Details

Annotation information of features with multiplicity larger than one is compressed as follows. It is assumed that all features map to the same chromosome, leaving no ambiguity. The start base pair of the "new" feature is the smallest start base pair of features from which it has been formed. The end base pair of the "new" feature is the largest end base pair of features from which it has been formed.

#### Value

Object of class `ExpressionSet`, restricted to the specified subset of features.
**Warning**

The phenoData, experimentData, and other slots of the ExpressionSet-object are currently not passed on to the subsetted object.

**Note**

This is a more intricate version of the ExpressionSet2subset function. They exist parallel because this function is much slower than its counterpart.

**Author(s)**

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

**References**


**See Also**

ExpressionSet2subset

**Examples**

```r
# load data
data(pollackGE16)

# extract genomic information from ExpressionSet-object
chr <- fData(pollackGE16)[,1]
bpstart <- fData(pollackGE16)[,2]
bpend <- fData(pollackGE16)[,3]

# find unique genomic locations
uniqInfo <- uniqGenomicInfo(chr, bpstart, bpend, verbose = FALSE)

# subset ExpressionSet-object to features with unique genomic locations
pollackGE16 <- ExpressionSet2weightedSubset(pollackGE16, uniqInfo, 1, 2, 3)
```

---

**getSegFeatures**  
Identical signature features selection from cghCall-object.

**Description**

Given an example, selects features (contiguous to the example) with the same signature (as the example) across samples from an cghCall-object.

**Usage**

```r
getSegFeatures(FeatureNo, CNdata, verbose=TRUE)
```
**hdEntropy**

**Arguments**
- `featureNo`: Row number of example feature.
- `CNdata`: Object of class `cghCall`.
- `verbose`: Logical indicator: should intermediate output be printed on the screen?

**Value**
Object of class numeric, containing the row numbers of those contiguous features with the same segmented log2-ratio signatures as `featureNo` across samples.

**Author(s)**
Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

**References**

**See Also**
cghCall, RCMestimation.

**Examples**
```r
# load data
data(pollackCN16)

# feature of interest
featureNo <- 7

# extract all features with identical copy number signature (over the samples)
getSegFeatures(featureNo, pollackCN16)
```

---

**hdEntropy**  
*Entropy estimation.*

**Description**
The (differential) entropy of a high-dimensional multivariate random variable is estimated from a (high-dimensional matrix) under a normality or k-NN distributional assumption.

**Usage**
hdEntropy(Y, method = "normal", k = 1, center = TRUE, indKnn = TRUE)
hdMI

Arguments

Y (High-dimensional) matrix. Rows are assumed to represent the samples, and columns represent the samples’ genes or traits.
method Distributional assumption under which entropy is to be estimated.
k k-nearest neighbor parameter.
center Logical indicator: should the columns of Y be centered around zero?
indKnn Logical indicator: should samples’ individual contributions to the k-NN entropy be reported?

Value

The entropy estimate is returned as a numeric.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

entropyTest.

Examples

data(pollackGE16)
hdEntropy(t(exprs(pollackGE16)), method="knn")

---

hdMI  

Mutual information estimation.

Description

The mutual information between two high-dimensional mutivariate random variables is estimated from two (high-dimensional matrix) under a normality or k-NN distributional assumption.

Usage

hdMI(Y, X, method = "normal", k = 1, center = TRUE, rescale = TRUE)
matchAnn2Ann

Arguments

\( Y \) (High-dimensional) matrix. Rows are assumed to represent the samples, and columns represent the samples’ genes or traits.

\( X \) (High-dimensional) matrix. Rows are assumed to represent the samples, and columns represent the samples’ genes or traits. The number of rows of \( X \) must be identical to that of \( Y \).

method Distributional assumption under which mutual information is to be estimated.

\( k \) k-nearest neighbor parameter.

center Logical indicator: should the columns (traits) of \( Y \) and \( X \) be centered at zero? Applied only under the normality assumption.

rescale Logical indicator: should \( Y \) and \( X \) be rescaled to have the same scale? Applied only under the k-NN assumption.

Value

The mutual information estimate is returned as a numeric.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

mutInfTest.

Examples

data(pollackCN16)
data(pollackGE16)
hdMI(t(exprs(pollackGE16)), t(copynumber(pollackCN16)), method="knn")

-------

matchAnn2Ann Genomic location matching of two sets of features

Description

Genomic location matching of two sets of features

Usage

matchAnn2Ann(chr1, bpstart1, bpend1, chr2, bpstart2, bpend2, method = "distance", maxDist = 10000, minPerc = 0, reference = 1, ncpus = 1, verbose=TRUE)
Arguments

chr1 Object of class numeric containing chromosome information of features from set 1.
bpstart1 Object of class numeric containing start base pair information of features from set 1. Of same length as chr1.
bpend1 Object of class numeric containing end base pair information of features from set 1. Of same length as chr1.
chr2 Object of class numeric containing chromosome information of features from set 2.
bpstart2 Object of class numeric containing start base pair information of features from set 2. Of same length as chr2.
bpend2 Object of class numeric containing end base pair information of features from set 2. Of same length as chr2.
method Matching method to be applied, either "distance" or "overlap". See below for details.
maxDist Maximum number of bases two features are allowed to be separated for a match. Only used in combination with method="distance".
minPerc Minimum percentage of overlap between two features required for a match. Only used in combination with method="overlap".
reference Platform that is taken as a reference in the calculation of the percentage, should equal 1 or two, referring to the platform.
ncpus Number of cpus to be used in the computation.
verbose Logical indicator: should intermediate output be printed on the screen?

Details

The features of set 1 (chr1, bpstart1, bpend1) are matched to the features of set 2 (chr2, bpstart2, bpend2). That is, for every feature in set 2, features in set 1 are sought.

In case method="distance", the midpoint of set 1 and set 2 features are calculated and for each feature of set 2 all features of set 1 with midpoints not further than maxDist are selected. If there are no features in set 1 satisfying this criterion, the feature of set 2 that could not be matched is discarded.

If method="overlap", each feature of set 1 is matched to the feature of set 2 on the basis of the percentage of overlap. All features of set 1 with a percentage exceeding minPerc are selected. In case no feature in set 1 had any overlap with the features from set 2, the feature of set 2 that could not be matched is discarded.

Value

An object of class list. Each list item is a three-column matrix with the matched features information. The first column contains feature numbers of set 1 in the order as supplied. The second column contains feature numbers of set 2 in the order as supplied. Each row thus has two entries. The first entry contains the feature number of set 1 that has been matched to second entry, representing the feature number of set 2. The third column contains either the percentage of overlap (method="overlap") or the distance between the the midpoints of the two features (method="distance").
Warning

Base pair information of features from both sets should be on the same scale!

Features with incomplete annotation information are removed before matching. For clarity, they are not included in the object with matched features.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

matchCGHcall2ExpressionSet

Examples

```r
# load data
data(pollackCN16)
data(pollackGE16)

# extract genomic information from cghCall-object
chr1 <- fData(pollackCN16)[,1]
bpstart1 <- fData(pollackCN16)[,2]
bpend1 <- fData(pollackCN16)[,3]

# extract genomic information from ExpressionSet-object
chr2 <- fData(pollackGE16)[,1]
bpstart2 <- fData(pollackGE16)[,2]
bpend2 <- fData(pollackGE16)[,3]

# match features from both platforms
matchedFeatures <- matchAnn2Ann(chr1, bpstart1, bpend1, chr2, bpstart2, bpend2, method = "distance", maxDist = 10000)
```

matchCGHcall2ExpressionSet

*Genomic location matching of CN and GE data*

Description

Integrative CN-GE analysis requires the copy number data of all genes on the expression array to be available. `intCNEan.match` matches the features of the copy number platform to the genes of the expression array. This is done using their genomic locations on the basis of either proximity or overlap.
Usage

matchCGHcall2ExpressionSet(CNdata, GEdata, CNchr, CNbpstart, CNbpend, GEchr, GEbpstart, GEbpend, method = "distance", reference=1, ncpus=1, verbose=TRUE)

Arguments

- **CNdata**: Object of class `cghCall`, containing (among others) annotation and call probabilities.
- **GEdata**: Object of class `ExpressionSet`.
- **CNchr**: Column in the slot `featureData` of the `cghCall`-object specifying the chromosome information of the features.
- **CNbpstart**: Column in the slot `featureData` of the `cghCall`-object specifying the start basepair information of the features.
- **CNbpend**: Column in the slot `featureData` of the `cghCall`-object specifying the end basepair information of the features.
- **GEchr**: Column in the slot `featureData` of the `ExpressionSet`-object specifying the chromosome information of the features.
- **GEbpstart**: Column in the slot `featureData` of the `ExpressionSet`-object specifying the start basepair information of the features.
- **GEbpend**: Column in the slot `featureData` of the `ExpressionSet`-object specifying the end basepair information of the features.
- **method**: Matching method to be applied, either "distance", "overlap" or "overlapPlus". See below for details.
- **reference**: Platform that is taken as a reference in the calculation of the percentage, should equal 1 or two, referring to the platform.
- **ncpus**: Number of cpus to be used in the computation.
- **verbose**: Logical indicator: should intermediate output be printed on the screen?

Details

Ideally full annotation information (chromosome number, start base pair, end base pair) for both copy number and gene expression data is available. In case only start base pair information is available, let CNbpstart and GEbpstart refer to the same columns as CNbpend and GEbpend. Base pair information of copy number and expression data should be on the same scale.

Matching occurs on the basis of genomic locations. In case `method="distance"`, the midpoint of CN and GE features are calculated and for each gene on the expression array the closest feature of the copy number platform is selected. If `method="overlap"`, each gene in the `ExpressionSet`-object is matched to the feature from the copy number platform with the maximum percentage of overlap. If the maximum percentage of overlap equals zero, the gene is not included in the matched objects. If `method="overlapPlus"`, the features are first matched by their percentage of overlap (as with the `method="overlap"`-option). For all non-matched GE features its closest two CN features (one down- and one upstream) are determined. If the copy number signature of these two CN features is identical, intrapolation seems reasonable, and and the GE feature is matched to the closest of these two CN features. Hence, `method="overlapPlus"` makes use of the copy number data, consequently, matching may be different for different data sets.
merge2cghCalls

Value

A two-column matrix with the matched features entries. The first column contains feature numbers of the cghCall-object. The second column contains feature numbers of the ExpressionSet-object. Each row thus has two entries. The first entry contains the feature number of the cghCall-object that has been matched to second entry, representing the feature number of the ExpressionSet-object.

Warning

Features with incomplete annotation information are removed before matching. For clarity, they are not included in the objects with matched features.

Note

The matching process implemented here is different from the one implemented in the (depreciated) ACEit-package (Van Wieringen et al., 2006).

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also
cghCall, ExpressionSet

Examples

# load data
data(pollackCN16)
data(pollackGE16)

# match features from both platforms
featureMatch <- matchCGHcall2ExpressionSet(pollackCN16, pollackGE16, 1, 2, 3, 1, 2, 3)

merge2cghCalls                  Merge two cghCall-objects into one cghCall-object

Description

Merge two cghCall-objects into one cghCall-object.

Usage

merge2cghCalls(CNdata1, CNdata2, verbose=TRUE)
Arguments

CNdata1 Object of class `cghCall`.
CNdata2 Object of class `cghCall`.
verbose Logical indicator: should intermediate output be printed on the screen?

Details

Data of the two objects is assumed to originate from the same samples, and are presented in the same order.

Only the experimental data and annotation information is inherited by the merged object.

Value

Object of class `cghCall`, restricted to the specified subset of features.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

cghCall.

Examples

```r
# load data
data(pollackCN16)

# create two cghCall-objects
ids1 <- sample(1:dim(pollackCN16)[1], 10)
CNdata1 <- pollackCN16[ids1,]
CNdata2 <- pollackCN16[-ids1,]

# order the copy number data genomically
pollackCN16 <- merge2cghCalls(CNdata1, CNdata2)
```
merge2ExpressionSets Merger two ExpressionSet-objects into one ExpressionSet-object

Description

Merge two ExpressionSet-objects into one ExpressionSet-object

Usage

merge2ExpressionSets(GEdata1, GEdata2, verbose=TRUE)

Arguments

GEdata1 Object of class ExpressionSet.
GEdata2 Object of class ExpressionSet.
verbose Logical indicator: should intermediate output be printed on the screen?

Details

Data of the two objects is assumed to originate from the same samples, and are presented in the same order.

Only the experimental data and annotation information is inherited by the merged object.

Value

Object of class ExpressionSet, restricted to the specified subset of features.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

ExpressionSet.
Examples

```r
# load data
data(pollackGE16)

# create two cghCall-objects
ids1 <- sample(1:dim(pollackGE16)[1], 10)
GEdata1 <- pollackGE16[ids1,]
GEdata2 <- pollackGE16[-ids1,]

# order the copy number data genomically
pollackGE16 <- merge2ExpressionSets(GEdata1, GEdata2)
```

---

**miTest-class**

Class "miTest" for storing the results of the function `mutInfTest`.

**Description**

The class `miTest` is the output of a call to `mutInfTest`. It stores results from a hypothesis test.

**Slots**

- `statistic`: Object of class "numeric". Observed test statistic (i.e., estimated mutual information).
- `p.value`: Object of class "numeric". P-value for the mutual information test.
- `null.dist`: Object of class "numeric". The permutation null distribution for the test statistic.
- `nperm`: Object of class "numeric". Number of permutation used for p-value calculation.
- `remark`: Object of class "character". Tells whether the permutation algorithm was terminated prematurely or not.

**Methods**

- **summary** signature(object = "miTest"): Prints the test results.

**Author(s)**

Wessel van Wieringen: <w.vanwieringen@vumc.nl>

**See Also**

- `mutInfTest`

**Examples**

```r
showClass("miTest")
```
mutInfTest

Test for mutual information

Description

A test evaluates the significance of the mutual information between two (high-dimensional) multivariate random variables. The null distribution is obtained via an efficient permutation resampling algorithm.

Usage

mutInfTest(Y, X, nPerm = 1000, method = "normal", k = 1, center = TRUE, rescale = TRUE, lowCiThres=0.10, ncpus=1, verbose=FALSE)

Arguments

Y  (High-dimensional) matrix. Columns are assumed to represent the samples, and rows represent the samples' genes or traits.
X  (High-dimensional) matrix. Columns are assumed to represent the samples, and rows represent the samples' genes or traits. The number of columns of X must be identical to that of Y.
nPerm  Number of permutations.
method  Distributional assumption under which mutual information is to be estimated.
k  k-nearest neighbor parameter.
center  Logical indicator: should the rows of Y and X be centered at zero? Applied only under the normality assumption.
rescale  Logical indicator: should Y and X be rescaled to have the same scale? Applied only under the k-NN assumption.
lowCiThres  A value between 0 and 1. Determines speed of efficient p-value calculation. If the probability of a p-value being below lowCiThres is smaller than 0.001 (read: the test is unlikely to become significant), the permutation analysis is terminated and a p-value of 1.00 is reported.
cpus  Number of cpus used for the permutations.
verbose  Logical indicator: should intermediate output be printed on the screen?

Value

Object of miTest-class.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References

**nBreakpoints**

**See Also**

hdMI

**Examples**

```r
# load data
data(pollackCN16)
data(pollackGE16)
Y <- t(exprs(pollackGE16))
X <- t(copynumber(pollackCN16))

# perform testing and print test results
testRes <- mutInfTest(Y, X, nPerm = 1000)
summary(testRes)
```

<table>
<thead>
<tr>
<th>nBreakpoints</th>
<th>Number of breakpoints</th>
</tr>
</thead>
</table>

**Description**

The number of samples with at least one breakpoint is calculated for each transcript.

**Usage**

```r
nBreakpoints(featuresAndWeights, CNdata)
```

**Arguments**

- `featuresAndWeights` Object of class `list`. Each list item is a `matrix`. The first column of this `matrix` contains the row numbers of features to be maintained in subsetting of the `cghCall`-object. The second column contains the weights of each features, to be used in the calculation of the weighted average copy number signal.

- `CNdata` Object of class `cghCall`

**Details**

For each item of the object `featuresAndWeights` the segmented data from the `cghCall`-object is used to determine whether a sample exhibits a breakpoint for this transcript.

**Value**

Object of class `numeric` containing the number of samples with at least one breakpoint. It is of the same length as the `featuresAndWeights`-object.

**Author(s)**

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>
References


See Also

matchAnn2Ann.

Examples

# load data
data(pollackCN16)
data(pollackGE16)

# extract genomic information from cghCall-object
chr1 <- fData(pollackCN16)[,1]
bpstart1 <- fData(pollackCN16)[,2]
bpend1 <- fData(pollackCN16)[,3]

# extract genomic information from ExpressionSet-object
chr2 <- fData(pollackGE16)[,1]
bpstart2 <- fData(pollackGE16)[,2]
bpend2 <- fData(pollackGE16)[,3]

# match features from both platforms
matchedIDs <- matchAnn2Ann(chr1, bpstart1, bpend1, chr2, bpstart2, bpend2, method = "distance", maxDist = 10000)

# extract ids for object subsetting
matchedIDsCN <- lapply(matchedIDs, function(Z){ return(Z[, -1, drop=FALSE]) })

# calculate the number of breakpoints
nBreakpoints(matchedIDsCN, pollackCN16)

# Penalized estimation of a pathway's regulatory network from DNA copy number and gene expression data (one-sample).

description

The regulatory relationships between DNA copy number and gene expression within a pathway are modeled by a simultaneous-equations model. Parameters of this model are fitted by minimizing of a penalized least squares criterion. The employed penalty is that of the lasso, encouraging sparsity.

Usage

pathway1sample(Y, X, lambda1 = 1, constr = TRUE, startCis=numeric(), startTrans=matrix(), verbose = FALSE)
Arguments

\(Y\) **matrix.** Rows are assumed to represent the samples, and columns represent the samples’ gene expression levels.

\(X\) **matrix.** Rows are assumed to represent the samples, and columns represent the samples’ genes or traits. The number of rows and columns of \(X\) must be identical to that of \(Y\).

\(\lambda_1\) **numeric or matrix.** The lasso parameter. In case \(\lambda_1\) is of class numeric and its length equals one, the same penalty parameter is applied to all trans-effects. In case \(\lambda_1\) is of class matrix its column and row dimension equal the number of columns of \(Y\). A possibly different penalty parameter is applied to each trans-effect.

\(\text{constr}\) **logical.** Should the cis-effect (the direct effect of a column of \(X\) on column of \(Y\)) be positive?

\(\text{startCis}\) **numeric.** Starting values for the cis-effect.

\(\text{startTrans}\) **matrix.** Starting values for the trans-effect.

\(\text{verbose}\) **logical.** Should intermediate output be printed on the screen?

Details

The model is fitted equation-by-equation. This is warranted by the assumption of independent errors. The expression levels of one gene is regressed on its own DNA copy number data and the expression levels of all other genes in the pathway.

Value

Object of class \text{pathwayFit}.

Author(s)

Wessel N. van Wieringen: \text{<w.vanwieringen@vumc.nl>}

References


See Also

See also \text{pathwayFit} and \text{pathway2sample}.

Examples

```r
# set number of genes (p) and samples (n)
p <- 10
n <- 1000

# sample cis-effects
beta <- abs(rnorm(p))

# sample trans-effects
Theta <- matrix(sample(c(-1,1), p^2, replace=TRUE, prob=c(0.2, 0.8)), ncol=p) *
       matrix(runif(p^2), ncol=p) / 4
```
# sample error variances
Sigma <- diag(rchisq(p, df=1)/5 + 0.5)

# sample DNA copy number data
X <- matrix(runif(n*p, min=-2, max=2), ncol=p)

# sample gene expression data
Y <- t(apply(X, 1, function(Y, beta){ Y * beta }, beta=beta)) %*% t(solve(Theta)) + rmvnorm(n, sigma=solve(Theta) %*% Sigma %*% t(solve(Theta)))

# fit model
pFit <- pathway2sample(Y, X, lambda1=1, verbose=TRUE)

# compare fit to "truth" for cis-effects
plot(pFit@Cis ~ beta, pch=20)

# compare fit to "truth" for trans-effects
penFits <- c(pFit@Trans[upper.tri(Theta)], pFit@Trans[lower.tri(Theta)])
truth <- c(Theta[upper.tri(Theta)], Theta[lower.tri(Theta)])
plot(penFits ~ truth, pch=20)

---

pathway2sample  
Penalized estimation of a pathway's regulatory network from DNA copy number and gene expression data (two-sample).

Description

The regulatory relationships between DNA copy number and gene expression within a pathway are modeled by a simultaneous-equations model. Parameters of this model are fitted by minimizing of a penalized least squares criterion. The employed penalty is a combination of the lasso and the fused lasso. This combination encourages within-sample sparsity (lasso), and limits the between-sample differences (fused lasso).

Usage

pathway2sample(Y, X, id, lambda1 = 1, lambdaF = 1, method = "FL", constr = TRUE, startCis = numeric(), startTrans1 = matrix(), startTrans2 = matrix(), epsilon = 0, verbose = FALSE)

Arguments

Y  
Object of class matrix. Rows are assumed to represent the samples, and columns represent the samples’ gene expression levels.

X  
Object of class matrix. Rows are assumed to represent the samples, and columns represent the samples’ genes or traits. The number of rows and columns of X must be identical to that of Y.

id  
An indicator variable of class numeric for the two groups to be compared. The groups should be coded as 0 and 1.
lambda1 Either a numeric- or matrix-object. The lasso parameter. In case lambda1 is of class numeric its length is one, and the same penalty parameter is applied to all trans-effects. In case lambda1 is of class matrix its column and row dimension equal the number of columns of Y. A possibly different penalty parameter is applied to each trans-effect.

lambdaF Either a numeric- or matrix-object. The fused lasso parameter. In case lambdaF is of class numeric and of length one, the same penalty parameter is applied to all differential trans-effects. In case lambdaF is of class matrix its column and row dimension equal the number of columns of Y. A possibly different penalty parameter is applied to each differential trans-effect.

method A character-object. Indicates which penalty to employ (see details).

constr logical. Should the cis-effect (the direct effect of a column of X on column of Y) be positive?

startCis numeric. Starting values for the cis-effect.

startTrans1 matrix. Starting values for the trans-effect of group 1 (coded as 0).

startTrans2 matrix. Starting values for the trans-effect of group 2 (coded as 1).

epsilon A numeric. Non-negative positive in the low-dimensional case. epsilon is to assume a positive value in the high-dimensional case.

verbose logical. Should intermediate output be printed on the screen?

Details

The model is fitted equation-by-equation. This is warranted by the assumption of independent errors. The expression levels of one gene is regressed on its own DNA copy number data and the expression levels of all other genes in the pathway.

The method-option indicates which penalty is combined with the least squares loss function. In case method = FL, this the fused lasso penalty (as described in Van Wieringen, W.N., Van de Wiel, M.A., 2012):

$$\lambda_1 \| \Theta^{(a)} \|_1 + \lambda_1 \| \Theta^{(b)} \|_1 + \lambda_F \| \Theta^{(a)} - \Theta^{(b)} \|_1.$$  

When method = FLs, this penalty is simplified to:

$$\lambda_1 \| \Theta^{(a)} + \Theta^{(b)} \|_1 + \lambda_F \| \Theta^{(a)} - \Theta^{(b)} \|_1.$$

The use of this penalty may be motivated as follows. The two samples used to share a common network architecture. One expects only a relatively limited number of edges to have changed. Hence, the majority of edges will have the same sign, resulting in equality of the two penalties. An other motivation for this second penalty arises from the the observation that it is computationally faster. And, as

$$\lambda_1 \| \Theta^{(a)} \|_1 + \lambda_1 \| \Theta^{(b)} \|_1 \geq \lambda_1 \| \Theta^{(a)} + \Theta^{(b)} \|_1,$$

it penalizes less. As such, the resulting FLs penalized estimates may be used as starting values for fitting the model with the FL penalty.

Value

Object of class pathwayFit.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>
References


See Also

See also pathwayFit and pathway1sample.

Examples

# set number of genes (p) and samples (n)
p <- 10
n <- 1000

# sample cis-effects
beta <- abs(rnorm(p))

# sample trans-effects for first sample
Theta1 <- matrix(sample(c(-1,1), p^2, replace=TRUE, prob=c(0.2, 0.8)), ncol=p) * matrix(runif(p^2), ncol=p) / 4
diag(Theta1) <- 1

# sample trans-effects for second sample
idDiff <- sample(which(Theta1 != 1), 10)
Theta2 <- Theta1[upper.tri(Theta1)] <- -Theta1[idDiff]

# sample error variances
Sigma <- diag(rchisq(p, df=1)/5 + 0.5)

# sample DNA copy number data of sample 1
X1 <- matrix(runif(n*p, min=-2, max=2), ncol=p)

# sample gene expression data
Y1 <- t(apply(X1, 1, function(Y, beta){ Y * beta }, beta=beta)) %*% t(solve(Theta1)) + rmvnorm(n, sigma=solve(Theta1) %*% Sigma %*% t(solve(Theta1)))

# sample DNA copy number data of sample 2
X2 <- matrix(runif(n*p, min=-2, max=2), ncol=p)

# sample gene expression data
Y2 <- t(apply(X2, 1, function(Y, beta){ Y * beta }, beta=beta)) %*% t(solve(Theta2)) + rmvnorm(n, sigma=solve(Theta2) %*% Sigma %*% t(solve(Theta2)))

# construct id-vector
id <- c(rep(0, n), rep(1, n))

# fit model
pFit <- pathway2sample(Y=rbind(Y1, Y2), X=rbind(X1, X2), id=id, lambda1=0, lambdaF=0.01)

# compare fit to "truth" for cis-effects
plot(pFit@Cis ~ beta, pch=20)

# compare fit to "truth" for differential trans-effects
penFits1 <- c(pFit@Trans1[upper.tri(Theta1)], pFit@Trans1[lower.tri(Theta1)])
penFits2 <- c(pFit@Trans2[upper.tri(Theta2)], pFit@Trans2[lower.tri(Theta2)])
pathwayFit-class

Class "pathwayFit" for storing the results of the function pathway1sample or pathway2sample.

Description

The class pathwayFit is the output of a call to pathway1sample and pathway2sample. It stores results from fitting a simultaneous-equations model from DNA copy number and gene expression data.

Slots

Cis: Object of class "numeric". Vector of estimated cis-effect.

Trans: Object of class "matrix". Matrix containing the trans-effects (one-sample only).

Trans1: Object of class "matrix". Matrix containing the trans-effects of the first sample (two-sample only).

Trans2: Object of class "matrix". Matrix containing the trans-effects of the second sample (two-sample only).

Sigma: Object of class "numeric". Vector of estimated residual variances.

lambda1: Object of class "matrix". Lasso parameter(s) employed.

lambdaF: Object of class "matrix". Fused lasso parameter(s) employed.

constr: Object of class "logical". Indicator for parameter constraints on cis-effect.

epsilon: Object of class "numeric". Constant used for the stabilization of estimation in a high-dimensional context.

method: Object of class "character". Indicator for method used in model fitting.

Methods

pathwayPlot signature(object = "pathwayFit"): Plots the pathwayFit-object.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

See Also

See also pathway1sample and pathway2sample.

Examples

showClass("pathwayFit")
pathwayPlot

Plot of the pathway topology as reconstructed from DNA copy number and gene expression data (one-sample only).

Description

Plotting the topology of a pathway’s regulatory network as reconstructed from DNA copy number and gene expression data by the `pathway1sample`-function.

Usage

`pathwayPlot(pFit, directed = TRUE, tWidth = 1, cWidth = 1, geWidth=10, cnWidth=10, circleDist = 1.5, gNames = NULL, main = "", remove = FALSE)`

Arguments

- `pFit`: Object of class `pathwayFit` as returned by the function `pathway1sample`.
- `directed`: A logical indicating whether to plot directed or undirected `trans`-effects.
- `tWidth`: A numeric that scales the width of the `trans`-effect edges.
- `cWidth`: A numeric that scales the width of the `cis`-effect edges.
- `geWidth`: A numeric that scales the width of the gene expression nodes.
- `cnWidth`: A numeric that scales the width of the DNA copy number nodes.
- `circleDist`: A numeric that scales the inner and outer circle.
- `gNames`: A character containing the gene names to written inside the nodes.
- `main`: The character to be plotted as plot title.
- `remove`: A logical indicating whether to remove genes without `trans`-effects.

Author(s)

Wessel N. van Wieringen: `<w.vanwieringen@vumc.nl>`

References


See Also

See also `pathway1sample`.

Examples

```r
# set number of genes (p) and samples (n)
p <- 10
n <- 1000

# sample `cis`-effects
beta <- abs(rnorm(p))

# sample `trans`-effects
```
Theta <- matrix(sample(c(-1,1), p^2, replace=TRUE, prob=c(0.2, 0.8)), ncol=p) * matrix(runif(p^2), ncol=p) / 4
diag(Theta) <- 1

# sample error variances
Sigma <- diag(rchisq(p, df=1)/5 + 0.5)

# sample DNA copy number data
X <- matrix(runif(n*p, min=-2, max=2), ncol=p)

# sample gene expression data
Y <- t(apply(X, 1, function(Y, beta){ Y * beta }, beta=beta)) %*% t(solve(Theta)) + rmvnorm(n, sigma=solve(Theta) %*% Sigma %*% t(solve(Theta)))

# fit model
pFit <- pathway1sample(Y, X, lambda1=500)

# plot pathway topology
pathwayPlot(pFit, tWidth=5, cWidth=5)

---

pollackCN16

Breast cancer data (copy number)

Description

Copy number data of chromosome 16 the breast cancer data set. Called using CGHcall with default settings, contains 240 features and 41 samples.

Usage

data(pollackCN16)

Format

An object of class cghCall.

Source


References


Examples

data(pollackCN16)
pollackGE16  

*Breast cancer data (gene expression)*

**Description**
Gene expression data of chromosome 16 of the breast cancer data set; contains 240 features and 41 samples.

**Usage**
```r
data(pollackGE16)
```

**Format**
An object of class `ExpressionSet`.

**Source**

**Examples**
```r
data(pollackGE16)
```

**profilesPlot**  

*CN-GE profiles plot*

**Description**
Plots a sample’s copy number and gene expression data side-by-side. This visualizes the relation between CN and GE within an individual sample.

**Usage**
```r
profilesPlot(CNdata, GEdata, sampleNo, chr = 0, verbose=TRUE)
```

**Arguments**

- **CNdata**: Object of class `cghCall`, containing (among others) annotation and call probabilities. Features should be matched with those of the accompanying `ExpressionSet`-object (as may be done using the `matchCGHcall2ExpressionSet`-function).
- **GEdata**: Object of class `ExpressionSet`. Features should be matched with those of the accompanying `cghCall`-object (as may be done using the `matchCGHcall2ExpressionSet`-function).
- **sampleNo**: Sample number of sample to be plotted. Corresponds to the order in which samples appear the `CNdata`- and `GEdata`-objects.
- **chr**: Chromosome number for which the profiles are to be plotted. Default `chr=0` for whole genome plotting.
- **verbose**: Logical indicator: should intermediate output be printed on the screen?
**Details**

The blue lines in the gene expression profile plot are the median expressions of genes that map to the same copy number segment.

**Author(s)**

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

**See Also**

cghCall, ExpressionSet

**Examples**

```r
# load data
data(pollackCN16)
data(pollackGE16)

# plot CN and GE profiles alongside
data(pollackCN16, pollackGE16, 23, 16)
```

**RCMestimation**

*Fitting of the random coefficients model.*

**Description**

The parameters of the random coefficients model are estimated by means of the maximum likelihood method. The implemented maximum likelihood procedure has been optimized with respect to computational efficiency and memory usage.

**Usage**

```r
RCMestimation(Y, X, R, hypothesis = "H2", shrinkType = "none", 
estType = "normal", corType = "unif", maxNoIt = 100, 
minSuccDist = 0.005, verbose = FALSE)
```

**Arguments**

- **Y**: The matrix containing the (e.g., expression) data (number of columns equal to number of features, number of rows equal to number of samples).
- **X**: The design matrix (number of rows equal to number of samples, number of columns equal to number of covariates).
- **R**: The linear constraint matrix (number of columns equal to the number of covariates).
- **hypothesis**: The hypothesis under which the model is fitted: $H_0$ ($H_0 : R \beta = 0 \& \tau^2 = 0$), $H_1$ ($H_1 : R \beta >= 0 \& \tau^2 = 0$), $H_2$ ($H_2 : R \beta >= 0 \& \tau^2 >= 0$).
- **shrinkType**: The type of shrinkage to be applied to the error variances: none (shrinkage parameter is set equal to zero: no shrinkage), opt (shrinkage parameter is chosen to minimize the mean squared error criterion) or full (shrinkage parameter is set equal to one).
**estType**
Type of estimation, either *normal* (non-robust) or *robust*.

**corType**
Correlation structure to be used, either *unif* or *ar1*.

**maxNoIt**
Maximum number of iterations in the ML procedure.

**minSuccDist**
Minimum distance between estimates of two successive iterations to be achieved.

**verbose**
Logical indicator: should intermediate output be printed on the screen?

**Details**
Details on the type of random coefficients model that is actually fitted are specified in the reference below.

**Value**
Object of class `rcmFit`.

**Note**
In case a covariate for the intercept is included in the design matrix $X$ we strongly recommend the center, per feature, the data around zero.

**Author(s)**
Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

**References**

**See Also**
`RCMrandom`, `RCMtest`, `rcmTest`.

**Examples**
```r
# load data
data(pollackCN16)
data(pollackGE16)

# select features belonging to a region
ids <- getSegFeatures(20, pollackCN16)

# extract segmented log2 ratios of the region
X <- t(segmented(pollackCN16)[ids[1], , drop=FALSE])

# extract segmented log2 ratios of the region
Y <- exprs(pollackGE16)[ids,]

# center the expression data (row-wise)
Y <- t(Y - apply(Y, 1, mean))

# specify the linear constraint matrix
R <- matrix(1, nrow=1)
```
# fit the random coefficients model to the random data
RCMresults <- RCMestimation(Y, X, R)

rcmFit-class

Class "rcmFit" for storing the results of the function RCMestimation.

Description

The class rcmFit is the output of a call to RCMestimation. It stores results from fitting a random coefficients model.

Slots

betas: Object of class "numeric". Vector of estimated global regression coefficients for each of the covariates in the design matrix.

tau2s: Object of class "numeric". Vector of estimated regression coefficient variances for each of the covariates in the design matrix X.

sigma2s: Object of class "numeric". Vector of estimated error variances for all genes.

rho: Object of class "numeric". Estimated correlation parameter between the error of two contiguous features.

av.sigma2s: Object of class "numeric". Average of the unshrunk estimated error variances.

shrinkage: Object of class "numeric". Applied shrinkage parameters in fitting the model.

loglik: Object of class "numeric". The log-likelihood of the fitted model.

corType: Object of class "character". Correlation structure of the error used.

X: Object of class "matrix". The design matrix.

Methods

RCMloss signature(object = "rcmFit"): Calculates the log-likelihood associated with the fitted model.

RCMrandom signature(object = "rcmFit"): Samples from the distribution induced by the fitted model.

summary signature(object = "rcmFit"): Prints the estimation result.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

See Also

RCMestimation, RCMrandom.

Examples

showClass("rcmFit")
Random data from the random coefficients model.

Description
The significance of hypotheses regarding parameters of the random coefficients model is assessed by means of the parametric bootstrap. Here random data from the fitted model under the null hypothesis of interest are drawn. This function provides.

Usage
RCMrandom(object)

Arguments
object Object of class rcmFit.

Details
Details on the type of random coefficients model from which data are drawn are specified in the reference below.

Value
A matrix of dimension (number of genes) times (number of samples).

Author(s)
Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References

See Also
RCMestimation, rcmFit.

Examples
# load data
data(pollackCN16)
data(pollackGE16)

# select features belonging to a region
ids <- getSegFeatures(20, pollackCN16)

# extract segmented log2 ratios of the region
X <- t(segmented(pollackCN16)[ids[1], , drop=FALSE])

# extract segmented log2 ratios of the region
Y <- exprs(pollackGE16)[ids,]

# center the expression data (row-wise)
Y <- t(Y - apply(Y, 1, mean))

# specify the linear constraint matrix
R <- matrix(1, nrow=1)

# fit the random coefficients model to the random data
RCMresults <- RCMestimation(Y, X, R)

# draw random data
Yrandom <- RCMrandom(RCMresults)

---

### Methods for Function `RCMrandom`

**Description**
Methods for function `RCMrandom`

**Methods**

- **signature(object = "rcmFit")** Draws random data of same dimension as data on which the `rcmFit`-object was fitted.

---

### `RCMtest`

**Hypothesis testing within the random coefficient model.**

**Description**
Function that evaluates various hypothesis within the random coefficients model via bootstrap re-sampling.

**Usage**

```r
RCMtest(Y, X, R, testType = "I", nBoot = 100, lowCiThres = 0.1, shrinkType = "none", estType = "normal", corType = "unif", maxNoIt = 100, minSuccDist = 0.005, returnNullDist = FALSE, ncpus=1, verbose = FALSE)
```

**Arguments**

- `Y` The matrix containing the (e.g., expression) data (number of columns equal to number of features, number of rows equal to number of samples).
- `X` The design matrix (number of rows equal to number of samples, number of columns equal to number of covariates).
- `R` The linear constraint matrix (number of columns equal to the number of covariates).
The hypothesis to be tested: I (H0 : R beta = 0 & tau2 = 0) vs. (H2 : R beta >= 0 V tau2 >= 0), II (H0 : R beta = 0 & tau2 = 0) vs. (H1 : R beta >= 0 & tau2 = 0), III (H1 : R beta >= 0 & tau2 = 0) vs. (H2 : R beta >= 0 & tau2 >= 0).

Number of bootstraps.

A value between 0 and 1. Determines speed of efficient p-value calculation. If the probability of a p-value being below lowCiThres is smaller than 0.001 (read: the test is unlikely to become significant), bootstrapping is terminated and a p-value of 1.00 is reported.

The type of shrinkage to be applied to the error variances: none (shrinkage parameter is set equal to zero: no shrinkage), opt (shrinkage parameter is chosen to minimize the mean squared error criterion) or full (shrinkage parameter is set equal to one).

Type of estimation, either normal (non-robust) or robust.

Correlation structure to be used, either unif or ar1.

Maximum number of iterations in the ML procedure.

Minimum distance between estimates of two successive iterations to be achieved.

Logical indicator: should the null distribution be returned?

Number of cpus used for the bootstrap.

Logical indicator: should intermediate output be printed on the screen?

Details

Details on the type of random coefficients model that is actually fitted are specified in the reference below.

Value

Object of class rcmTest.

Warning

In case a covariate for the intercept is included in the design matrix X we strongly recommend the center, per feature, the data around zero.

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

RCMestimation, RCMrandom, rcmTest.
Examples

# load data
data(pollackCN16)
data(pollackGE16)

# select features belonging to a region
ids <- getSegFeatures(20, pollackCN16)

# extract segmented log2 ratios of the region
X <- t(segmented(pollackCN16)[ids[1], drop=FALSE])

# extract segmented log2 ratios of the region
Y <- exprs(pollackGE16)[ids,]

# center the expression data (row-wise)
Y <- t(Y - apply(Y, 1, mean))

# specify the linear constraint matrix
R <- matrix(1, nrow=1)

# fit the random coefficients model to the random data
RCMresults <- RCMestimation(Y, X, R)

# test for significance of effect of X on Y
RCMtestResults <- RCMtest(Y, X, R, nBoot=2)
summary(RCMtestResults)

rcmTest-class

Class "rcmTest" for storing the results of the function RCMtest.

Description

The class rcmTest is the output of a call to RCMtest. It stores results from a hypothesis test.

Slots

statistic: Object of class "numeric". Observed test statistic (i.e., estimated mutual information).
p.value: Object of class "numeric". P-value for the mutual information test.
betas: Object of class "numeric". Vector of estimated global regression coefficients for each of the covariates in the design matrix.
tau2s: Object of class "numeric". Vector of estimated regression coefficient variances for each of the covariates in the design matrix.
sigma2s: Object of class "numeric". Vector of estimated error variances for all features.
rho: Object of class "numeric". Estimated correlation parameter between the error of two contiguous features.
av.sigma2s: Object of class "numeric". Average of the unshrunken estimated error variances.
shrinkage: Object of class "numeric". Type of shrinkage applied in the estimation.
loglik: Object of class "numeric". The log-likelihood of the fitted model.
nBoot: Object of class "numeric". Number of bootstraps used for p-value calculation.
splitMatchingAtBreakpoints

split matching at breakpoints

corType: Object of class "character". Correlation structure used in the fitted model.
null.dist: Object of class "numeric". The permutation null distribution for the test statistic.
remark: Object of class "character". Tells whether the bootstrapping was terminated prematurely or not.

Methods

summary signature(object = "rcmTest"): Prints the test results.

Author(s)

Wessel van Wieringen: w.vanwieringen@vumc.nl

See Also

RCMtest

Examples

showClass("rcmTest")

splitMatchingAtBreakpoints

Description

In case a feature of platform 1 has been matched to multiple features of another platform, instead of averaging the data from these features, one may consider splitting the data at breakpoints within genes. This function modifies the results from the matching function matchAnn2Ann to facilitate this. The result can than directly be used in the subsetting functions cghCall2weightedSubset and ExpressionSet2weightedSubset.

Usage

splitMatchingAtBreakpoints(matchedIDs, CNdata)

Arguments

matchedIDs An object of class list, as returned by the matchAnn2Ann-function.
CNdata Object of class cghCall.

Value

An object of class list, similar to that returned by the matchAnn2Ann-function.

Author(s)

Wessel N. van Wieringen: w.vanwieringen@vumc.nl
References


See Also

`matchAnn2Ann, cghCall2weightedSubset, ExpressionSet2weightedSubset`.

Examples

```r
# load data
data(pollackCN16)
data(pollackGE16)

# extract genomic information from cghCall-object
chr1 <- fData(pollackCN16)[,1]
bpstart1 <- fData(pollackCN16)[,2]
bpend1 <- fData(pollackCN16)[,3]

# extract genomic information from ExpressionSet-object
chr2 <- fData(pollackGE16)[,1]
bpstart2 <- fData(pollackGE16)[,2]
bpend2 <- fData(pollackGE16)[,3]

# match features from both platforms
matchedFeatures <- matchAnn2Ann(chr1, bpstart1, bpend1, chr2, bpstart2, bpend2, method = "distance", maxDist = 10000)

# expand
matchedFeatures <- splitMatchingAtBreakpoints(matchedFeatures, pollackCN16)
```

---

**summary-method**  

**Methods for Function summary**

Description

Methods for function summary

Methods

signature(object = "ANY") Regular.  
signature(object = "entTest") Print output.  
signature(object = "miTest") Print output.  
signature(object = "rcmFit") Print output.  
signature(object = "rcmTest") Print output.
uniqGenomicInfo

Description

Finds unique genomic location information.

Usage

uniqGenomicInfo(chr, bpstart, bpend, verbose = FALSE)

Arguments

- **chr**: Object of class `numeric` containing chromosome information of features.
- **bpstart**: Object of class `numeric` containing start base pair information of features. Of same length as `chr`.
- **bpend**: Object of class `numeric` containing end base pair information of features. Of same length as `chr`.
- **verbose**: Logical indicator: should intermediate output be printed on the screen?

Value

An object of class `list`. Each list item is a four-column matrix with the matched features information. The first column contains feature numbers of features with identical genomic location. The second, third and fourth column contain the chromosome, start and end base pair information of the features (should be the same for each feature).

Author(s)

Wessel N. van Wieringen: <w.vanwieringen@vumc.nl>

References


See Also

ExpressionSet2weightedSubset, cghCall2weightedSubset

Examples

```r
# load data
data(pollackGE16)

# extract genomic information from ExpressionSet-object
chr <- fData(pollackGE16)[,1]
bpstart <- fData(pollackGE16)[,2]
bpend <- fData(pollackGE16)[,3]

# find unique genomic locations
uniqInfo <- uniqGenomicInfo(chr, bpstart, bpend, verbose = FALSE)
```
Index

*Topic classes
  cisTest-class, 17
  entTest-class, 20
  miTest-class, 35
  pathwayFit-class, 43
  rcmFit-class, 49
  rcmTest-class, 53

*Topic datasets
  pollackCN16, 45
  pollackGE16, 46

*Topic methods
  RCMrandom-method, 51
  summary-method, 55

*Topic package
  sigaR-package, 2
  .RCM1loss-method, 3

  cghCall, 4–8, 25, 26, 31–33, 37, 45–47, 54
  cghCall2cghSeg, 4
  cghCall2maximumSubset, 5
  cghCall2order, 6
  cghCall2subset, 7
  cghCall2weightedSubset, 8, 22, 55
  cghSeg, 4, 9–11
  cghSeg2order, 9
  cghSeg2subset, 10
  cghSeg2weightedSubset, 11
  cisEffectPlot, 12
  cisEffectTable, 13
  cisEffectTest, 13, 14, 14, 17, 18
  cisEffectTune, 15, 16
  cisTest-class, 17
  CNGEheatmaps, 18
  entropyTest, 19, 20, 27
  entTest-class, 20
  expandMatching2singleIDs, 21
  ExpressionSet, 13, 15, 17, 22–24, 31, 32, 34, 46, 47
  ExpressionSet2order, 22
  ExpressionSet2subset, 23
  ExpressionSet2weightedSubset, 22, 24, 55
  getSegFeatures, 25
  hdEntropy, 20, 26
  hdMI, 27, 37
  matchAnn2Ann, 6, 22, 28, 38, 55
  matchCGHcall2ExpressionSet, 30, 46
  merge2cghCalls, 32
  merge2ExpressionSets, 34
  miTest-class, 35
  mutInfTest, 28, 35, 36
  nBreakpoints, 37
  pathway1sample, 38
  pathway2sample, 40
  pathwayFit-class, 43
  pathwayPlot, 44
  pollackCN16, 45
  pollackGE16, 46
  profilesPlot, 46
  RCMestimation, 26, 47, 49, 50, 52
  rcmFit-class, 49
  RCMrandom, 48, 49, 50, 52
  RCMrandom, rcmFit-method
    (RCMrandom-method), 51
  RCMrandom-method, 51
  RCMtest, 48, 51, 53, 54
  rcmTest-class, 53
  sigaR(sigaR-package), 2
  sigaR-package, 2
  splitMatchingAtBreakpoints, 54
  summary, ANY-method (summary-method), 55
  summary, entTest-method
    (summary-method), 55
  summary, miTest-method (summary-method), 55
  summary, rcmFit-method (summary-method), 55
  summary, rcmTest-method
    (summary-method), 55
  summary-method, 55
  uniqGenomicInfo, 56