Package 'SIM'

December 5, 2025

Type Package

Title Integrated Analysis on two human genomic datasets

Version 1.81.0

Author Renee X. de Menezes and Judith M. Boer

Maintainer Renee X. de Menezes <r.menezes@nki.nl>

Imports graphics, stats, globaltest, quantsmooth

Depends R (>= 3.5), quantreg

Suggests biomaRt, RColorBrewer

Description Finds associations between two human genomic datasets.

License GPL (>= 2)

biocViews Microarray, Visualization

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

git_branch devel

git_last_commit ca66dc5

git_last_commit_date 2025-10-29

Repository Bioconductor 3.23

Date/Publication 2025-12-04

Contents

`SIM-package` *Statistical Integration of Microarrays*

Description

SIM is a statistical model to identify associations between two genomic datasets. Where one is assigned as dependent variable and the other as independent e.g. copy number measurements on several samples versus expression measurements on the same samples. A region of interest can be chosen to run the integrated analysis on either the same region for both dependent and independent datasets or different regions. For each dependent feature a P-value measures the association with the independent data, the contribution of each independent feature is given as Z-scores. The integrated analysis is based on the random-effect model for gene-sets as implemented in gt.

maybe something about annotation?

By default we use `method.adjust = "BY"` (Benjamini-Yekutieli) for multiple testing correction. This method accounts for dependence between measurements and is more conservative than "BH" (Benjamini-Hochberg). For details on the multiple testing correction methods see p.adjust. We have experienced that a rather low stringency cut-off on the BY-values of 20% allows the detection of associations for data with a low number of samples or a low frequency of abberations. False positives are rarely observed.

Make sure that the array probes are mapped to the same builds of the genome, and that the chrom.table used by the integrated.analysis is from the same build as well. See sim.update.chrom.table.

Details

Package:	SIM
Type:	Package
Version:	1.7.1
Date:	2010-09-14
License:	Open

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

References

Menezes RX, Boetzer M, Sieswerda M, van Ommen GJ, Boer JM (2009). Integrated analysis of DNA copy number and gene expression microarray data using gene sets. *BMC Bioinformatics*, **10**, 203-.

Goeman JJ, van de Geer SA, de Kort F, van Houwelingen HC (2004). A global test for groups of genes: testing association with a clinical outcome. *Bioinformatics*, **20**, 93-109.

See Also

assemble.data, integrated.analysis, sim.plot.zscore.heatmap, sim.plot.pvals.on.region, sim.plot.pvals.on.genome, tabulate.pvals, tabulate.top.dep.features, tabulate.top.indep.features, impute.nas.by.surrounding, sim.update.chrom.table, sim.plot.overlapping.indep.dep.features, getoverlappingregions

Examples

```
#load the datasets and the samples to run the integrated analysis
data(expr.data)
data(acgh.data)
data(samples)

#assemble the data
assemble.data(dep.data = acgh.data,
              indep.data = expr.data,
              dep.ann = colnames(acgh.data)[1:4],
              indep.ann = colnames(expr.data)[1:4],
              dep.id="ID",
              dep.chr = "CHROMOSOME",
              dep.pos = "STARTPOS",
              dep.symb="Symbol",
              indep.id="ID",
              indep.chr = "CHROMOSOME",
              indep.pos = "STARTPOS",
              indep.symb="Symbol",
              overwrite = TRUE,
              run.name = "chr8q")

#run the integrated analysis
integrated.analysis(samples = samples,
                    input.regions ="8q",
                    zscores=TRUE,
                    run.name = "chr8q")

# use functions to plot the results of the integrated analysis

#plot the P-values along the genome
sim.plot.pvals.on.genome(input.regions = "8q",
                         significance = c(0.2, 0.05),
```

```
                        adjust.method = "BY",
                        pdf = FALSE,
                        run.name = "chr8q")

#plot the P-values along the regions
sim.plot.pvals.on.region(input.regions = "8q",
       adjust.method="BY",
       run.name = "chr8q")

#plot the z-scores in an association heatmap
#plot the zscores in a heatmap
sim.plot.zscore.heatmap(input.regions = "8q",
                        method="full",
                        significance=0.2,
                        z.threshold=3,
                        show.names.indep=TRUE,
                        show.names.dep=TRUE,
                        adjust.method = "BY",
                        add.plot = "smooth",
                        smooth.lambda = 2,
                        pdf = FALSE,
                        run.name = "chr8q")

sim.plot.zscore.heatmap(input.regions = "8q",
                        method="full",
                        significance = 0.05,
                        z.threshold = 1,
                        show.names.indep=TRUE,
                        show.names.dep=FALSE,
                        adjust.method = "BY",
                        add.plot = "heatmap",
                        smooth.lambda = 2,
                        pdf = FALSE,
                        run.name = "chr8q")

sim.plot.zscore.heatmap(input.regions = "8q",
                        method="full",
                        significance = 0.05,
                        z.threshold = 1,
                        show.names.indep=TRUE,
                        show.names.dep=TRUE,
                        adjust.method = "BY",
                        add.plot = "none",
                        pdf = FALSE,
                        run.name = "chr8q")

#tabulate the P-values per region (prints to screen)
tabulate.pvals(input.regions = "8q",
               adjust.method="BY",
               bins=c(0.001,0.005,0.01,0.025,0.05,0.075,0.10,0.20,1.0),
               run.name = "chr8q")

table.dep <- tabulate.top.dep.features(input.regions="8q",
```

```
                    adjust.method="BY",
        method="full",
        significance=0.05,
        run.name="chr8q")
head(table.dep[["8q"]])

table.indep <- tabulate.top.indep.features(input.regions="8q",
                                    adjust.method="BY",
            method="full",
            significance= 0.05,
            z.threshold=c(-1, 1),
            run.name="chr8q")
head(table.indep[["8q"]])

sim.plot.overlapping.indep.dep.features(input.regions="8q",
                                  adjust.method="BY",
          significance=0.1,
          z.threshold= c(-1,1),
          log=TRUE,
          summarize="consecutive",
          pdf=FALSE,
          method="full",
          run.name="chr8q")
```

---

acgh.data *Array Comparative Genomic Hybridization data*

Description

Copy number data taken from Pollack et al. PNAS. 2002, 99(20): 12963-8.

Usage

```
data(acgh.data)
```

Format

A data frame with 99 observations on 45 variables. The first 4 columns are the unique identifier, symbol for the chromosome and start position of the probe the next 41 columns are the copy number measurements of 41 samples.

Details

A subset of the original data is taken namely all the probes on the long arm of chromosome 8.

Source

http://genome-www.stanford.edu/aCGH_breast/data.shtml

References

Pollack JR, Sorlie T, Perou CM, Rees CA, Jeffrey SS, Lonning PE, Tibshirani R, Botstein D, Borresen-Dale AL, Brown PO (2002). Microarray analysis reveals a major direct role of DNA copy number alteration in the transcriptional program of human breast tumors. *Proc Natl Acad Sci USA*. **1**, 99(20), 12963-8.

Examples

```
data(acgh.data)
```

`assemble.data` *Assemble the data to run the integrated analysis*

Description

Assembles the dependent and independent data and annotation of the both data sets.

Usage

```
assemble.data(dep.data,
  indep.data,
  dep.id = "ID",
  dep.chr = "CHROMOSOME",
  dep.pos = "STARTPOS",
  dep.ann = NULL,
  dep.symb,
  indep.id = "ID",
  indep.chr = "CHROMOSOME",
  indep.pos = "STARTPOS",
  indep.ann = NULL,
  indep.symb,
  overwrite = FALSE,
  run.name = "analysis_results")
```

Arguments

`dep.data`	The dependent data (`data.frame`), along with annotations. Each row should correspond to one feature. The following columns are expected to exist, and the column names should be inserted in the function. `dep.id`: A unique identifier. `dep.chr`: The number of the chromosome (1,2, ..., 22, X, Y). `dep.pos`: The base pair position, relative to the chromosome. `dep.symb`: Gene symbol (optional). `dep.ann`: Annotation can be multiple columns.
`indep.data`	`data.frame` The independent data, along with annotations. Each row should correspond to one feature. The following columns are expected to exist, and the column names should be inserted in the function. `indep.id`: A unique identifier. `indep.chr`: The number of the chromosome (1,2, ..., 22, X, Y). `indep.pos`: The base pair position, relative to the chromosome. `indep.symb`: Gene symbol (optional). `indep.ann`: Annotation can be multiple columns.

dep.ann	vector with either the names of the columns or the column numbers in the dependent data that contain the annotation.
indep.ann	vector with either the names of the columns or the column numbers in the independent data that contain the annotation.
dep.id	vector with the column name in the dependent data that contains the ID. Will be used in the sim.plot.zscore.heatmap function. Empty ID's will be substituted by NA.
dep.chr	vector with column name in the dependent data that contains the chromosome numbers.
dep.pos	vector with the column name in the dependent data that contains the position on the chromosome in bases.
dep.symb	Optional, either missing or a single vector with the column name in the dependent data that contains the symbols. Will be used in sim.plot.zscore.heatmap as label.
indep.id	vector with the column name in the independent data that contains the ID. Will be used in the sim.plot.zscore.heatmap function. Empty ID's will be substituted by NA.
indep.chr	vector with the column name in the independent data that contains the chromosome numbers.
indep.pos	vector with the column name in the independent data that contains the position on the chromosome in bases.
indep.symb	Optional, either missing or a vector with the column name in the dependent data that contains the Symbols. Will be used in sim.plot.zscore.heatmap as label.
overwrite	logical, indicate when a run.name is already present, the results can be overwritten.
run.name	Name of the analysis. The results will be stored in a folder with this name in the current working directory (use getwd() to print the current working directory). If the missing, the default folder "analysis_results" will be generated.

Details

Based on the chromosome and probe position an absolute position is calculated according to $chromosomenumber * 1e9 + probeposition$. Chromosome column is converted to factor and releveled according to the levels of the chrom.table, so the only levels allowed are c(1:22, "X", "Y"). Currently only human genome support without mitochondrial DNA.

Value

No values are returned. Instead, the datasets and annotation columns are stored in separate files in the data folder in the directory specified in run.name. If assemble.data has run succesfully, the integrated.analysis function can be performed.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

See Also

SIM, integrated.analysis

Examples

```
# Generate datasets and the samples to run the integrated analysis
set.seed(53245)
ngenes <- 100
nsamples <- 100
# generate copy number measurements
x <- matrix(rnorm(n = ngenes*nsamples), nrow = ngenes, ncol = nsamples)
# add mean shift effect for half of the samples, copy gain for 2nd half of the genes
x[ seq_len(ngenes/2), seq_len(nsamples/2)] <- x[ seq_len(ngenes/2), seq_len(nsamples/2)] + 2
# generate gene expression with normal distribution and mean equal to gene copy number
y <- rnorm(n = ngenes*nsamples, mean = matrix(x, nrow = ngenes*nsamples, ncol = 1), sd = 0.8)
y <-  matrix(y, nrow = ngenes, ncol = nsamples)
samples <- paste0("S", seq_len(nsamples))
colnames(x) <- colnames(y) <- samples
# Making data objects
acgh.data <- data.frame(ID = paste0("G", seq_len(ngenes)),
                    CHROMOSOME = rep(1, ngenes),
                    STARTPOS = seq_len(ngenes)*12*10^5,
                    Symbol = paste0("Gene", seq_len(ngenes)),
                    x)
expr.data <- data.frame(ID = paste0("G", seq_len(ngenes)),
                       CHROMOSOME = rep(1, ngenes),
                       STARTPOS = seq_len(ngenes)*12*10^5,
                       Symbol = paste0("Gene", seq_len(ngenes)),
                       y)

#assemble the data
assemble.data(dep.data = acgh.data,
              indep.data = expr.data,
              dep.ann = colnames(acgh.data)[1:4],
              indep.ann = colnames(expr.data)[1:4],
              dep.id="ID",
              dep.chr = "CHROMOSOME",
              dep.pos = "STARTPOS",
              dep.symb="Symbol",
              indep.id="ID",
              indep.chr = "CHROMOSOME",
              indep.pos = "STARTPOS",
              indep.symb="Symbol",
              overwrite = TRUE,
              run.name = "chr1p")
```

`chrom.table` *Table with chromosome information*

Description

A table indicating the base positions of the beginning and end of chromosome arms and bands. Currently based on the UCSC March 2006/NCBI 36 build of the human genome.

Usage

```
data(chrom.table)
```

Format

A data frame with 862 observations on the following 6 variables.

chr, arm, band, start, end, stain

Details

Possibly the `chrom.table` can be update by sim.update.chrom.table. Currently only human genome support without mitochondrial DNA.

See Also

sim.update.chrom.table

Examples

```
data(chrom.table)
```

---

expr.data *Expression data example*

Description

Expression data taken from Pollack et al. PNAS. 2002, 99(20): 12963-8.

Usage

```
data(expr.data)
```

Format

A data frame with 99 observations on 45 variables. The first 4 columns are the unique identifier, symbol for the chromosome and start position of the probe the next 41 columns are the expression log-ratios of 41 samples.

Details

A subset of the original data is taken namely all the probes on the long arm of chromosome 8.

Source

http://genome-www.stanford.edu/aCGH_breast/data.shtml

References

Pollack JR, Sorlie T, Perou CM, Rees CA, Jeffrey SS, Lonning PE, Tibshirani R, Botstein D, Borresen-Dale AL, Brown PO (2002). Microarray analysis reveals a major direct role of DNA copy number alteration in the transcriptional program of human breast tumors. *Proc Natl Acad Sci USA*. **1**, 99(20), 12963-8.

Examples

```
data(expr.data)
```

---

getoverlappingregions *Get the overlapping regions between independent and dependent regions*

Description

Generates a table with overlapping regions.

Usage

```
getoverlappingregions(independent_regions,
                      dependent_regions,
                      method = c("union", "overlapping"))
```

Arguments

independent_regions	data.frame().Independent regions found with tab tabulate.top.indep.features.
dependent_regions	data.frame().Independent regions found with tab tabulate.top.dep.features.
method	method to estimate the overlapping regions, either "union" or "overlapping". overlapping outputs only the overlapping parts of the overlapping regions. union outputs the whole region. Say independent region = 1-10, dependent region = 5-12. The output is 1-12.

Details

Calculates the overlap between two tables.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

See Also

SIM, tabulate.top.dep.features, tabulate.top.indep.features, sim.plot.overlapping.indep.dep.features

Examples

```
#no examples yet!
```

`impute.nas.by.surrounding` *Impute NA's in array-CGH data*

Description

Replace an NA by the median of the surrounding features in the same sample.

Usage

```
impute.nas.by.surrounding(dataset,
                          window.size = 5)
```

Arguments

`dataset`	data.frame with the dataset to replace the NA's by the medians of the surrounding features.
`window.size`	numeric value, specifying of how many features around the NA the median should be taken.

Details

This function can be used when the dependent dataset in the integrated.analysis function is array-CGH data and contains probes that have an NA. To avoid loosing data by throwing away the probes with NA's, the impute.nas.by.surrounding function can be used which simply takes the median of the probes around an NA. The number of probes used for the imputatin is chosen by giving a value for window.size. This script takes quite long to run!

Value

A `data.frame` is returned, containing the inserted "dataset" all `NA` replaced with median of the window of size "window.size" around the `NA`.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes `<R.X.Menezes@lumc.nl>`

See Also

SIM, assemble.data, integrated.analysis

Examples

```
#no examples yet!
```

`integrated.analysis` *Integrated analysis of dependent and indepedent microarray data*

Description

Runs the Integrated Analysis to test for associations between dependent and independent microarray data on the same set of samples.

Usage

```
integrated.analysis(samples,
                            input.regions = "all chrs",
                            input.region.indep = NULL,
                            zscores =  FALSE,
                            method = c("full", "smooth", "window", "overlap"),
                            dep.end = 1e5,
                            window = c(1e6, 1e6),
                            smooth.lambda=2,
                            adjust = ~1,
                            run.name = "analysis_results",
                            ...)
```

Arguments

`samples`	`vector` with either the names of the columns in the dependent and independent data corresponding to the samples, or a numerical vector containing the column numbers to include in the analysis, e.g. 5:10 means columns 5 till 10. Make sure that both datasets have the same number of samples with the same column names!
`input.regions`	`vector` indicating the dependent regions to be analyzed. Can be defined in four ways: `1) predefined input region:` insert a predefined input region, choices are: "all chrs", "all chrs auto", "all arms", "all arms auto" In the predefined regions "all arms" and "all arms auto" the arms 13p, 14p, 15p, 21p and 22p are left out, because in most studies there are no or few probes in these regions. To include them, just make your own vector of arms. `2) whole chromosome(s):` insert a single chromosome or a list of chromosomes as a `vector: c(1, 2, 3). 3) chromosome arms:` insert a single chromosome arm or a list of chromosome arms like `c("1q", "2p", "2q"). 4) subregions of a chromosome:` insert a chromosome number followed by the start and end position like `"chr1:1-1000000"` These regions can also be combined, e.g. `c("chr1:1-1000000","2q", 3)`. See `details` for more information.

`input.region.indep`
: indicating the independent region which will be analysed in combination of the dependent region. Only one input region can given using the same format as the dependent input region.

`zscores`
: `logical` indicates whether the Z-scores are calculated (takes longer time to run). If `zscores = FALSE`, only P-values are calculated.

`method`
: either one of "full", "window", "overlap" or "smooth". This defines how the data is used for the`integrated.analysis`. full: the whole dependent data region is taken. window: takes the middle of the dependent probe and does the integration on the independent probes that are within the window given at window-size given by `window`. overlap: does the integration on the independent probes that are within the start and end of the dependent probes given at `dep.end`. smooth: does smooth on the dependent probes with smoothing factor given at `smooth.lambda`, finds the value of smooth for each independent probe and does the integration on them. Only needed when `method = "smooth"`, default `smooth.lambda = 2`

`dep.end`
: `numeric` or `character` either the name of the column "end" in the dependent data or, when not available, an numeric value which indicates the end deviating from the start. When a numeric value is inserted, the function will do: $start + dep.end = end$. Only needed when `method = "window"` or "overlap".

`window`
: numeric values. Window to search for overlapping independent features per dependent probe. First value is the number of positions to the left from the middle of the probe, the second value is the number of positions to the right from the middle of the probe. Only needed when `method = "window"`.

`smooth.lambda`
: `numeric` factor used for smoothing the dependent data. Only needed when `method = "smooth"`. See quantsmooth for more information. By default the `segment = min(nrow(dep.data), 100)`.

`adjust`
: `formula` a formula like ~gender, where gender is a vector of the same size as samples. The regression models is correct for the gender effect, see gt.

`run.name`
: `character` name of the analysis. The results will be stored in a folder with this name in the current working directory (use `getwd()` to print the current working directory). If missing the default folder `"analysis\_results"` will be generated.

`...`
: additional arguments for gt e.g. `model="logistic"` or when `permutations > 0` the null distribution is estimated using permutations, see gt. See Details.

Details

The Integrated Analysis is a regression of the independent data on the dependent features. The regression itself is done using the gt, which means that the genes in a region (e.g. a chromosome arm) are tested as a gene set. The individual associations between each dependent and each independent feature are calculated as Z-scores (standardized influences, see `?gt`).

This function splits the datasets into separate sets for each region (as specified by the `input.regions`) and runs the analysis for each region separately.

When running the Integrated Analysis for a predefined input region, like "all arms" and "all chrs", output can be obtained for all input regions, as well as subsets of it. But note that the genomic

unit must be the same: if `integrated.analysis` was run using chromosomes as units, any of the functions and plots must also use chromosomes as units, and not chromosome arms. Similarly, if `integrated analysis` was run using chromosome arms as units, these units must also be used to produce plots and outputs. For example if the `input.regions = "all arms"` was used, P-value plots (see sim.plot.pvals.on.region can be produced by inserting the `input.regions = "all arms"`, but also for instance "1p" or "20q". However, to produce a plot of the whole chromosome, for example chromosome 1, the integrated should be re-run with `input.region=1`. The same goes for "all chrs": P-value plots etc. can be produced for chromosome 1,2 and so on... but to produce plots for an arm, the `integrated.analysis` should be re-run for that region. This also goes for subregions of the chromosome like "chr1:1-1000000".

By default the gt uses a "linear" model, only when the dependent data is a `logical matrix` containing `TRUE` and `FALSE` a "logistic" model is selected. All other models need `model = ""`, see gt for available models.

Value

No values are returned. Instead, the results of the analysis are stored in the subdirectories of the directory specified in `run.name`. E.g. the z-score matrices are saved in subfolder `method`.

The following functions can be used to visualize the data:

1) sim.plot.zscore.heatmap (only possible when `zscores = TRUE`)
2) sim.plot.pvals.on.region
3) sim.plot.pvals.on.genome
4) sim.plot.overlapping.indep.dep.features

Other functions can be used to tabulate the results:

1) tabulate.pvals
2) tabulate.top.dep.features
3) tabulate.top.indep.features (only possible when `zscores = TRUE`
4) getoverlappingregions (only possible when tablulate.top.dep.features and tabulate.top.indep.features were run.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

References

Menezes RX, Boetzer M, Sieswerda M, van Ommen GJ, Boer JM (2009). Integrated analysis of DNA copy number and gene expression microarray data using gene sets. *BMC Bioinformatics*, **10**, 203-.

Goeman JJ, van de Geer SA, de Kort F, van Houwelingen HC (2004). A global test for groups of genes: testing association with a clinical outcome. *Bioinformatics*, **20**, 93-109.

See Also

SIM, sim.plot.zscore.heatmap, sim.plot.pvals.on.region, sim.plot.pvals.on.genome, tabulate.pvals, tabulate.top.dep.features, tabulate.top.indep.features, getoverlappingregions, sim.plot.overlapping.indep.dep.features, gt

Examples

```
#first run example(assemble.data)
data(samples)
#perform integrated analysis without Z-scores using the method = "full"
integrated.analysis(samples=samples,
     input.regions="8q",
     zscores=FALSE,
     method="full",
     run.name="chr8q")
```

`link.metadata` *Link a metadata annotation file to expression ID*

Description

Get annotation out of a AnnotationData package and link them to the expression data using the expression probe ID's

Usage

```
link.metadata(data = expr.data,
              col.ID.link = 1,
              chr = as.list(hgu133plus2CHR),
              chrloc = as.list(hgu133plus2CHRLOC),
              symbol = as.list(hgu133plus2SYMBOL))
```

Arguments

`data`	`data.frame` with expression data including an expression probe ID column.
`col.ID.link`	numeric value, specifying the column of `data` that contains the ID to link with the `poslist`.
`chr`	`list` specifying the metadata annotation of the chromosome location on the genome.
`chrloc`	`list` specifying the metadata annotation of the location of the probe on the chromosome.
`symbol`	`list` specifying the metadata annotation of the symbol corresponding to the probe.

Details

Often, the annotation for expression array probes lack chromosome position information. Therefore, this function adds the two required columns to run the integrated.analysis: "CHROMOSOME" and "STARTPOS". In addition, the optional column, "Symbol" is added.

Value

A `data.frame` is returned, containing a dataset with annotation columns which can be used forintegrated.analysis.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee x Menezes <R.X.Menezes@lumc.nl>

See Also

RESOURCERER.annotation.to.ID

Examples

```
# first download and install the AnnotationData package for your expression array platform
# for example
## Not run: library(hgu133plus2)
## Not run: expr.data <- link.metadata(data, col.ID.link = 1, chr = as.list(hgu133plus2CHR),
chrloc = as.list(hgu133plus2CHRLOC), symbol = as.list(hgu133plus2SYMBOL))
## End(Not run)
```

RESOURCERER.annotation.to.ID

Link RESOURCERER annotation file to expression ID

Description

Get annotation out of the RESOURCERER annotation file and link them to expression data with help of expression ID's

Usage

```
RESOURCERER.annotation.to.ID(data, poslist, col.ID.link = 1, col.poslist.link = 1)
```

Arguments

data	data.frame with expression data including an expression ID column.
poslist	data.frame containing the RESOURCERER annotation file
col.ID.link	numeric value, specifying the column of data that contains the ID to link with the poslist.
col.poslist.link	numeric value, specifying the column of poslist that contains the ID to link with the data.

Details

This function will output the inserted dataset, including the necessary, for integrated.analysis, a nnotation columns: "CHROMOSOME", "STARTPOS"and "Symbol" out of the inserted RESOURCE RER annotation file poslist.

Value

A `data.frame` is returned, containing a dataset with annotation columns which can be used for integrated.analysis

Author(s)

Marten Boetzer, Melle Sieswerda, Renee x Menezes <R.X.Menezes@lumc.nl>

See Also

link.metadata

Examples

```
# download expression array annotation from RESOURCERER
# ftp://occams.dfci.harvard.edu/pub/bio/tgi/data/Resourcerer
# it may be necessary to remove the first row, which states the genome build used for mapping
## Not run: read.an <- read.delim("affy_U133Plus2.txt", sep="\t", header=T)

# get physical mapping columns
## Not run: expr.data <- RESOURCERER_annotation_to_ID(data = read.expr, poslist = read.an, col.ID.link = 1, col.pos
```

`samples` *Samples for example data*

Description

Vector of sample names corresponding to the column headers containing the data in both the copy number (`acgh.data`) and expression (`expr.data`) example datasets.

Usage

```
data(samples)
```

Format

A character vector.

Source

http://genome-www.stanford.edu/aCGH_breast/data.shtml

References

Pollack JR, Sorlie T, Perou CM, Rees CA, Jeffrey SS, Lonning PE, Tibshirani R, Botstein D, Borresen-Dale AL, Brown PO (2002). Microarray analysis reveals a major direct role of DNA copy number alteration in the transcriptional program of human breast tumors. *Proc Natl Acad Sci USA*. **1**, 99(20), 12963-8.

Examples

```
data(samples)
```

`sim.plot.overlapping.indep.dep.features` *P-value plot and mean-zscore plots with indication of overlapping features.*

Description

Generates three plots: The first plot contains the P-values along the region, with the cut-off displayed. The second plot contains the mean-zscores along the region, with the cut-offs displayed. The third plots generates the cytobands of the region.

Usage

```
sim.plot.overlapping.indep.dep.features(input.regions,
input.region.indep = NULL,
adjust.method = "BY",
log = FALSE,
significance = 0.2,
max.pow = 5,
z.threshold = c(-3,3),
summarize = c("consecutive", "stretch", "window"),
stretch = 10,
window = 1e6,
percentage = 0.5,
xlim=NULL,
pdf = FALSE,
method = c("full", "smooth", "window", "overlap"),
run.name = "analysis_results", ...)
```

Arguments

`input.regions` `vector` indicating the dependent regions to be analyzed. Can be defined in four ways: `1) predefined input region:` insert a predefined input region, choices are: "all chrs", "all chrs auto", "all arms", "all arms auto" In the predefined regions "all arms" and "all arms auto" the arms 13p, 14p, 15p, 21p and 22p are left out, because in most studies there are no or few probes in these regions. To include them, just make your own vector of arms. `2) whole chromosome(s):` insert a single chromosome or a list of chromosomes as a `vector: c(1, 2, 3). 3) chromosome arms:` insert a single chromosome arm or a list of chromosome arms like `c("1q", "2p", "2q"). 4) subregions of a chromosome:` insert a chromosome number followed by the start and end position like `"chr1:1-1000000"` These regions can also be combined, e.g. `c("chr1:1-1000000","2q", 3)`. See `integrated.analysis` for more information.

`input.region.indep`
: indicating the independent region which will be analysed in combination of the dependent region. Only one input region can given using the same format as the dependent input region.

`adjust.method`
: Method used to adjust the P-values for multiple testing, see p.adjust. Default is "BY" recommended when copy number is used as dependent data. See SIM for more information about adjusting P-values.

`log`
: `logical` default `log = FALSE`, if `log = TRUE` P-values are plotting on $log[10]$ scale.

`significance`
: The threshold for selecting significant P-values.

`max.pow`
: `numeric` only when `log = TRUE` scale of the y-axis.

`z.threshold`
: Threshold to display a green or red bar in the color bar on top of the heatmap for independent features with mean z-scores above `z.threshold` (high positive association) or below `-z.threshold` (high negative association).

`summarize`
: either one of "consecutive", "stretch", "window" which visualizes the subregions according to the selected summarization a) "consecutive" shows the consecutive significant regions b) "stretch" shows a percentage `percentage` significant stretch of size `stretch`

and c) "window" a percentage `percentage` significant window of length `window`

`stretch`
: `integer` length of stretch, default `stretch = 10`

`window`
: `integer` length of window, default `window = 1e6`

`percentage`
: `numeric` a number between $[0, 1]$ given the percentage of significance in either the "stretch" of "window"

`xlim`
: `c(min, max)` scale of the x-axis. Can be used for zooming in on a region.

`pdf`
: `logical` indicate whether to generate a pdf of the plots in the current working directory or not.

`method`
: this must be the either full, window, overlap or smooth but the data should generated by the same method in `integrated.analysis`.

`run.name`
: This must be the same a given to `integrated.analysis`

`...`
: not used in this version

Details

details: Cytobands plot adapted from SNPChip

Value

No values are returned.The results are stored in a subdirectory of `run.name` as pdf.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

See Also

SIM, tabulate.top.dep.features, tabulate.top.indep.features, getoverlappingregions

Examples

```
#first run example(assemble.data)
#and example(integrated.analysis)
#overview plot of the dependent and independent features
sim.plot.overlapping.indep.dep.features(input.regions="8q",
                                adjust.method="BY",
          significance=0.1,
          z.threshold= c(-1,1),
          log=TRUE,
          summarize="consecutive",
          pdf=FALSE,
          method="full",
          run.name="chr8q")
```

`sim.plot.pvals.on.genome`

Plot the P-values in whole genome overview

Description

Generates a plot of the analyzed dependent data probe positions and their significance on all chromosomes.

Usage

```
sim.plot.pvals.on.genome(input.regions = "all chrs",
            significance = c(0.05, 0.20),
       adjust.method = "BY",
       method = c("full", "smooth", "window", "overlap"),
       run.name = "analysis_results",
       pdf = TRUE,
       main = "Significantly associated features",
       ylab = "Chromosomes",
       ann=par("ann"),
       ...)
```

Arguments

`input.regions`	`vector` indicating the dependent regions to be analyzed. Can be defined in four ways: `1)` `predefined` `input` `region:` insert a predefined input region, choices are: "all chrs", "all chrs auto", "all arms", "all arms auto" In the predefined regions "all arms" and "all arms auto" the arms 13p, 14p, 15p, 21p and 22p are left out, because in most studies there are no or few probes in these regions. To include them, just make your own vector of arms. `2)` `whole`

	chromosome(s): insert a single chromosome or a list of chromosomes as a vector: c(1, 2, 3). 3) chromosome arms: insert a single chromosome arm or a list of chromosome arms like c("1q", "2p", "2q"). 4) subregions of a chromosome: insert a chromosome number followed by the start and end position like "chr1:1-1000000" These regions can also be combined, e.g. c("chr1:1-1000000","2q", 3). See integrated.analysis for more information.
significance	Two values that categorize the P-values on the selected chromosomes. These margins are indicated by different colors shown in the legend. These values can be defined, e.g. pval.sig = c(0.3, 0.075)
adjust.method	Method used to adjust the P-values for multiple testing. see p.adjust Default is "BY" recommended when copy number is used as dependent data. See SIM for more information about adjusting P-values.
method	this must be the either full, window, overlap or smooth but the data should generated by the same method in integrated.analysis.
pdf	Boolean. Indicate whether to generate a pdf of the plots in the current working directory or not.
run.name	This must be the same a given to integrated.analysis
main	the usual graphical parameter for the caption of the plot.
ylab	the usual graphical parameter for the y-axis label of the plot.
ann	the usual graphical parameter for annotation of the plot.
...	Arguments to be passed to pdf when pdf = TRUE, see Details.

Details

Grey vertical lines indicate unsignificant probes on top the significant ones are plotted. A purple dot indicates the centromere and a organe line the input region.

Sometimes it is useful to make the genome-plot as A4 landscape-format, add the following parameters to the sim.plot.pvals.on.genome(..., paper='a4r', width=0, height=0)

Value

No values are returned. The results are stored in the folder "pvalue.plots" in directory run.name as pdf.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

See Also

SIM, sim.plot.zscore.heatmap, sim.plot.pvals.on.region

Examples

```
#first run example(assemble.data)
#and example(integrated.analysis)
#plot the p-values along the genome
sim.plot.pvals.on.genome(input.regions="8q",
                  significance=c(0.05, 0.005),
       adjust.method="BY",
       method="full",
       pdf=FALSE,
       run.name="chr8q")
```

`sim.plot.pvals.on.region`

P-value histograms and P-values along the genome per region

Description

Generates two plots of the P-values for an analyzed region. The first plot contains the distribution of the raw P-values and ranked plots of the raw and adjusted P-values. The second plot contains the P-values along the genome of analyzed input regions.

Usage

```
sim.plot.pvals.on.region(input.regions = c("all chrs"),
       significance = 0.2,
                        adjust.method = "BY",
                        method = c("full", "smooth", "window", "overlap"),
                        run.name = "analysis_results", ...)
```

Arguments

`input.regions`	`vector` indicating the dependent regions to be analyzed. Can be defined in four ways: `1) predefined input region:` insert a predefined input region, choices are: "all chrs", "all chrs auto", "all arms", "all arms auto" In the predefined regions "all arms" and "all arms auto" the arms 13p, 14p, 15p, 21p and 22p are left out, because in most studies there are no or few probes in these regions. To include them, just make your own vector of arms. `2) whole chromosome(s):` insert a single chromosome or a list of chromosomes as a `vector: c(1, 2, 3). 3) chromosome arms:` insert a single chromosome arm or a list of chromosome arms like `c("1q", "2p", "2q"). 4) subregions of a chromosome:` insert a chromosome number followed by the start and end position like `"chr1:1-1000000"` These regions can also be combined, e.g. `c("chr1:1-1000000","2q", 3)`. See `integrated.analysis` for more information.
`significance`	The threshold for selecting significant P-values.
`adjust.method`	Method used to adjust the P-values for multiple testing. see p.adjust Default is "BY" recommended when copy number is used as dependent data. See SIM for more information about adjusting P-values.

method	this must be the either full, window, overlap or smooth but the data should generated by the same method in integrated.analysis.
run.name	This must be the same a given to integrated.analysis
...	Arguments to be passed to pdf.

Details

This function returns a pdf containing the P-value plots. The second plot contains the multiple testing corrected P-values plotted along the chromosome (arm). On the x-axis, the start positions of the dependent features are displayed. On the y-axis, the P-value levels are displayed. Two dotted lines indicate P-value levels 0.2 and 0.1. In general, P-values below 0.2 are said to be "significant".

Value

No values are returned. The results are stored in a subdirectory of run.name as pdf.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

See Also

SIM, sim.plot.pvals.on.genome

Examples

```
#first run example(assemble.data)
#and example(integrated.analysis)
#plot the p-values along the regions
sim.plot.pvals.on.region(input.regions="8q",
                  adjust.method="BY",
       method="full",
       run.name="chr8q")
```

---

sim.plot.zoom.in *Zoom in on heatmap*

Description

Zoom in on pervious produced heatmap by sim.plot.zscore.heatmap

Usage

```
sim.plot.zoom.in(call)
```

Arguments

call	language, function call of sim.plot.zscore.heatmap

Details

sim.plot.zscore.heatmap returns (invisible) the function call and as attribute the local function environment. So by adjusting the the xlim and ylim a zoom in is created.

Value

an additional plot is created, on the previous plot a rectangle indicating the zoomed region.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

See Also

SIM, sim.plot.zscore.heatmap

Examples

```
#first run example(assemble.data)
#and example(integrated.analysis)

#plot the zscores in a heatmap
sim.plot <- sim.plot.zscore.heatmap(input.regions = "8q",
                                    adjust.method = "BY",
                                    run.name = "chr8q", pdf = FALSE)
#only when runned interactive
if(interactive())
 sim.plot.zoom.in(sim.plot)
```

`sim.plot.zscore.heatmap`

Association heatmap from z-scores

Description

Produces an association heatmap that shows the association (standardized influence) of each independent feature (expression measurement) with each dependent feature (copy number measurement). A P-value bar on the left indicates test signficance. A color bar on top indicates genes with mean z-scores across the signficant copy number probes above a set threshold. A summary of the copy number data helps to identify what copy number alterations are present in a region of association with expression. Positive association can mean copy number gain and increased expression, or deletion and decreased expression. The heatmaps can also be used in an exploratory analysis, looking for very local effects of copy number changes (usually small amplifications) on gene expression, that do not lead to a significant test result.

Usage

```
sim.plot.zscore.heatmap(input.regions = "all chrs",
   input.region.indep = NULL,
  method = c("full", "smooth", "window", "overlap"),
  adjust = ~1,
  significance = 0.2,
  z.threshold = 3,
  colRamp = colorRampPalette(c("red", "black", "green")),
  add.colRamp = colorRampPalette(c("blue", "black", "yellow"))(7),
  show.names.indep = FALSE,
  show.names.dep = FALSE,
  adjust.method = "BY",
  scale,
  add.scale,
  add.plot = c("smooth", "none", "heatmap"),
  smooth.lambda = 2,
  pdf = TRUE,
  run.name = "analysis_results",...)
```

Arguments

input.regions	vector indicating the dependent regions to be analyzed. Can be defined in four ways: 1) predefined input region: insert a predefined input region, choices are: "all chrs", "all chrs auto", "all arms", "all arms auto" In the predefined regions "all arms" and "all arms auto" the arms 13p, 14p, 15p, 21p and 22p are left out, because in most studies there are no or few probes in these regions. To include them, just make your own vector of arms. 2) whole chromosome(s): insert a single chromosome or a list of chromosomes as a vector: c(1, 2, 3). 3) chromosome arms: insert a single chromosome arm or a list of chromosome arms like c("1q", "2p", "2q"). 4) subregions of a chromosome: insert a chromosome number followed by the start and end position like "chr1:1-1000000" These regions can also be combined, e.g. c("chr1:1-1000000","2q", 3). See integrated.analysis for more information.
input.region.indep	indicating the independent region which will be analysed in combination of the dependent region. Only one input region can given using the same format as the dependent input region.
method	this must be the either full, window, overlap or smooth but the data should generated by the same method in integrated.analysis.
adjust	This variable must be a vector with the same length as samples or FALSE. The vector will be transformed to a factor and the levels of this will be coloured according to their subtype. When subtype=FALSE, all the samples will be coloured black.
significance	The threshold for selecting significant P-values.
z.threshold	Threshold to display a green or red bar in the color bar on top of the heatmap for independent features with mean z-scores above z.threshold (high positive association) or below -z.threshold (high negative association).

colRamp	Palette of colors to be used in the heatmap.
add.colRamp	Palette of colors to be used in the added plot.
show.names.indep	logical if set to TRUE, displays the names (indep.id and in dep.symb entered in the assemble.data) of the independent features with mean z-scores above or below the z.threshold in the heatmap.
show.names.dep	logical if set to TRUE, displays the names (dep.id and dep.sy mb entered in the assemble.data) of the significant dependent features in the heatmap.
adjust.method	Method used to adjust the P-values for multiple testing, see p.adjust. Default is "BY" recommended when copy number is used as dependent data. See SIM for more information about adjusting P-values.
scale	Vector specifying the color scale in the heatmap. If scale="auto", the maximum and minimum value of all z-scores will be calculated and set as the limits for all analyzed regions. Another option is to define a custom scale, e.g. scale = c(-5,5).
add.scale	Vector specifying the color scale in the left plot near the heatmap. If scale="auto", the maximum and minimum value of all the values will be calculated and set as the limits for all analyzed regions. Another option is to define a custom scale, e.g. scale = c(-5,5).
add.plot	Summary plot of copy number data in left panel. Either "smooth","heatmap", or "none". The "smooth" plot smoothes the copy number log ratios per sample, see quantsmooth for more details. The "heatmap" method produces an aCGH heatmap where green indicates gain, and red loss. The scale of the aCGH heatmap is automatically set to the min and max of the aCGH measurements of the analyzed regions. Default is plot.method = "none", no additional plot will be drawn.
smooth.lambda	Numeric value, specifying the quantile smoothing parameter for plot.method="smooth". See quantsmooth and references for more information.
pdf	logical; indicate whether to generate a pdf of the plots in the current working directory or not.
run.name	This must be the same a given to integrated.analysis
...	not used in this version

Details

The sim.plot.zscore.heatmap function can only run after the integrated.analysis is run with zscores = TRUE.

The results are returned as a single-page pdf containing an association heatmap of the regions listed in input.regions. For high-density arrays large files will be produced, both demanding more memory available from your computer to produce them as well as being heavier to open on screen. To avoid this, analyze chromosome arms as units instead of chromosomes, both here and in input.regions = "all arms".

The heatmap contains the z-scores generated by the function integrated.analysis with zscores=TRUE. The dependent features are plotted from bottom to top, the independent features from left to right. Positive associations are shown in green, negative associations in red (color scale on the right).

At the left side of the heatmap a color bar represents the multiple testing corrected P-values of the probes in the dependent data (copy number), also with a color legend. Dependening on which `plot.method` is used, a summary of copy number changes is shown on the left. At the top of the heatmap is a color bar corresponding to the mean z-scores of the independent features (expression data) that are above or below the `z.threshold`. If `show.names.indep` is set to TRUE, labels will be drawn for the probes with mean z-scores greater than `z.threshold` or lower than `-z.threshold` at the bottom of the heatmap. If `show.names.dep` is set to TRUE, labels will be drawn for the significant dependent probes lower than `significance` to the right of the heatmap.

Value

No values are returned. The results are stored in a subdirectory of `run.name` as pdf.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

References

Eilers PH, de Menezes RX. 2005 Apr 1, Quantile smoothing of array CGH data. *Bioinformatics*, **21**(7):1146-53.

Wang P, Kim Y, Pollack J, Narasimhan B, Tibshirani R. 2005, A method for calling gains and losses in array CGH data. *Biostatistics*, **6** :45-58.

See Also

SIM, tabulate.pvals, tabulate.top.dep.features, tabulate.top.indep.features, sim.plot.overlapping.indep.dep.features

Examples

```
#first run example(assemble.data)
#and example(integrated.analysis)

#plot the zscores in a heatmap
sim.plot.zscore.heatmap(input.regions = "8q",  adjust.method = "BY",
                        run.name = "chr8q", pdf = FALSE)

sim.plot.zscore.heatmap(input.regions = "8q",
                        method="full",
                        significance = 0.05,
                        z.threshold = 1,
                        colRamp = colorRampPalette(c("red", "black",
                           "green"))(15),
                        show.names.indep=TRUE,
                        show.names.dep=TRUE,
                        adjust.method = "holm",
                        add.plot = "heatmap",
                        smooth.lambda = 2,
                        pdf = FALSE,
                        run.name = "chr8q")
```

```
sim.plot.zscore.heatmap(input.regions = "8q",
                        method="full",
                        significance = 0.05,
                        z.threshold = 1,
                        show.names.indep = TRUE,
                        show.names.dep = TRUE,
                        add.plot = "none",
                        smooth.lambda = 2,
                        scale = c(-2, 2),
                        pdf = FALSE,
                        run.name = "chr8q")
```

`sim.update.chrom.table` *Update the chromomosome table*

Description

A function to update the genomic positions of chromosome arms. Base locations of the start and end of chromosome arms should be used from the same organism and build of genome as the location provided as annotation with the datasets.

Usage

```
sim.update.chrom.table(db = "homo_sapiens_core_40_36b")
```

Arguments

db	database name

Details

This functions requires library RMySQL. Currently SIM only supports integrated analysis on the human genome without mitochondrial DNA.

Value

Chromosome table chrom.table.

Author(s)

Marten Boetzer, Renee X. de Menezes <R.X.Menezes@lumc.nl>

References

http://www.ensembl.org/info/data/mysql.html

See Also

SIM, chrom.table

Examples

```
#youn need internet connection for this!
#sim.update.chrom.table(db = "homo_sapiens_core_40_36b")
```

`tabulate.pvals` *Sums significant P-values for the analyzed regions*

Description

Generates a `data.frame` with the significance of P-values in the analyzed regions, dividing them into bins.

Usage

```
tabulate.pvals(input.regions = "all chrs",
               adjust.method = "BY",
               bins = c(0.001, 0.005, 0.01, 0.025, 0.05, 0.075, 0.1, 0.2, 1),
               significance.idx = 8,
               order.by,
               decreasing = TRUE,
               method = c("full", "smooth", "window", "overlap"),
              run.name = "analysis_results")
```

Arguments

`input.regions` `vector` indicating the dependent regions to be analyzed. Can be defined in four ways: `1) predefined input region:` insert a predefined input region, choices are: "all chrs", "all chrs auto", "all arms", "all arms auto" In the predefined regions "all arms" and "all arms auto" the arms 13p, 14p, 15p, 21p and 22p are left out, because in most studies there are no or few probes in these regions. To include them, just make your own vector of arms. `2) whole chromosome(s):` insert a single chromosome or a list of chromosomes as a `vector: c(1, 2, 3). 3) chromosome arms:` insert a single chromosome arm or a list of chromosome arms like `c("1q", "2p", "2q"). 4) subregions of a chromosome:` insert a chromosome number followed by the start and end position like `"chr1:1-1000000"` These regions can also be combined, e.g. `c("chr1:1-1000000","2q", 3)`. See `integrated.analysis` for more information.

`adjust.method` Method used to adjust the P-values for multiple testing, see p.adjust. Default is "BY" recommended when copy number is used as dependent data. See SIM for more information about adjusting P-values.

`bins` `vector` of significance thresholds. This function will calculate the number of features having a P-value lower than the bin.

`significance.idx` Index of "bins" to use when computing the percentage of significant P-values. Defaults to 8 (i.e. the first entry in "bins"), in this case 0.20.

order.by	Column used for sorting the table. Defaults to "%" (i.e. the percentage of significant p-values).
decreasing	Direction used for sorting. Defaults to TRUE (i.e. highest values on top).
method	this must be the either full, window, overlap or smooth but the data should generated by the same method in integrated.analysis.
run.name	This must be the same a given to integrated.analysis

Value

Returns a data.frame. Each row corresponds to a chromosome and has as many entries as entries in bins, plus 1. Each entry contains the number of P-values that is smaller or equal to the corresponding entry in bins.

The last entry holds the percentage of P-values that is smaller than or equal to the bin identified by significance.idx.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

See Also

SIM, tabulate.top.dep.features, tabulate.top.indep.features

Examples

```
#first run example(assemble.data)
#and example(integrated.analysis)
tabulate.pvals(input.regions="8q",
         adjust.method="BY",
      bins=c(0.001,0.005,0.01,0.025,0.05,0.075,0.10,0.20,1.0),
      run.name="chr8q")
```

---

tabulate.top.dep.features

Lists the P-values for the dependent features

Description

Lists the integrated analysis P-values for the dependent features in the analyzed regions, together with the available annotation.

Usage

```
tabulate.top.dep.features(input.regions = "all chrs",
                          adjust.method="BY",
                          method = c("full", "smooth", "window", "overlap"),
                          significance = 1,
                          run.name = "analysis_results")
```

Arguments

input.regions	vector indicating the dependent regions to be analyzed. Can be defined in four ways: 1) predefined input region: insert a predefined input region, choices are: "all chrs", "all chrs auto", "all arms", "all arms auto" In the predefined regions "all arms" and "all arms auto" the arms 13p, 14p, 15p, 21p and 22p are left out, because in most studies there are no or few probes in these regions. To include them, just make your own vector of arms. 2) whole chromosome(s): insert a single chromosome or a list of chromosomes as a vector: c(1, 2, 3). 3) chromosome arms: insert a single chromosome arm or a list of chromosome arms like c("1q", "2p", "2q"). 4) subregions of a chromosome: insert a chromosome number followed by the start and end position like "chr1:1-1000000" These regions can also be combined, e.g. c("chr1:1-1000000","2q", 3). See integrated.analysis for more information.
adjust.method	Method used to adjust the P-values for multiple testing, see p.adjust. Default is "BY" recommended when copy number is used as dependent data. See SIM for more information about adjusting P-values.
method	this must be the either one of "full", "window", "overlap" or "smooth" but the data should generated by the same method in integrated.analysis.
significance	threshold used to select the significant dependent features. Pvalues below this threshold will be used to estimate regions.
run.name	This must be the same a given to integrated.analysis

Details

Output is a .txt file containing a table with sorted integrated analysis P-values of the dependent features. It includes the ann.dep columns that were read in the assemble.data function. Additionally it returns a .txt file containing the significant P-value rich regions. No P-value rich regions are returned when zscores.diag = "all".

Value

Returns a list of data.frame's for each input region. Significant P-value rich regions are returned as a data.frame. This data.frame can be used as an input for getoverlappingregions. Additionally, the results are stored in a subdirectory of run.name as txt.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

See Also

SIM, tabulate.pvals, tabulate.top.indep.features

Examples

```
#first run example(assemble.data)
#and example(integrated.analysis)
#get the top dependent features sorted by p-value
```

```
table.dep <- tabulate.top.dep.features(input.regions="8q",
                    adjust.method="BY",
        method="full",
        significance=0.05,
        run.name="chr8q")
head(table.dep[["8q"]])
```

`tabulate.top.indep.features` *Lists the mean z-scores for the independent features*

Description

Lists the mean z-scores for independent features in the analyzed regions, calculated across the significant dependent features. Gives insight in the expression levels most strongly associated with copy number changes.

Usage

```
tabulate.top.indep.features(input.regions = "all chrs",
       input.region.indep = NULL,
                            method = c("full", "smooth", "window", "overlap"),
                            adjust.method = "BY",
                            significance = 1,
                            decreasing=TRUE,
                            z.threshold = c(0, 0),
                            run.name = "analysis_results")
```

Arguments

`input.regions`	`vector` indicating the dependent regions to be analyzed. Can be defined in four ways: `1) predefined input region:` insert a predefined input region, choices are: "all chrs", "all chrs auto", "all arms", "all arms auto" In the predefined regions "all arms" and "all arms auto" the arms 13p, 14p, 15p, 21p and 22p are left out, because in most studies there are no or few probes in these regions. To include them, just make your own vector of arms. `2) whole chromosome(s):` insert a single chromosome or a list of chromosomes as a `vector: c(1, 2, 3). 3) chromosome arms:` insert a single chromosome arm or a list of chromosome arms like `c("1q", "2p", "2q"). 4) subregions of a chromosome:` insert a chromosome number followed by the start and end position like `"chr1:1-1000000"` These regions can also be combined, e.g. `c("chr1:1-1000000","2q", 3)`. See `integrated.analysis` for more information.
`input.region.indep`	fill in
`method`	this must be the either one of "full", "window", "overlap" or "smooth" but the data should generated by the same method in `integrated.analysis`.

adjust.method	Method used to adjust the P-values for multiple testing, see p.adjust. Default is "BY" recommended when copy number is used as dependent data. See SIM for more information about adjusting P-values.
significance	threshold used to select the significant dependent features. Only the z-scores with these features are used to calculate the mean z-scores across the independent features.
decreasing	fill in
z.threshold	fill in
run.name	This must be the same a given to integrated.analysis

Details

tabulate.top.indep.features can only be run after integrated.analysis with zscores = TRUE.

Output is a .txt file containing a table with the mean z-scores of all independent features per analyzed region. It includes the ann.indep columns that were read in the assemble.data function.

Additionally it returns a .txt file containing the significant zscores rich regions.

Depending on the argument "adjust.method", the P-values are first corrected for multiple testing. Next, th e z-scores are filtered to include only those entries that correspond to significant (P-value < "significa nce") dependent features to calculate the mean z-scores.

The dependent table can not be generated for diagonal integrated runs.

Value

Returns a list of data.frame's for each input region. Significant P-value rich regions are returned as a data.frame. This data.frame can be used as an input for getoverlappingregions. Additionally, the results are stored in a subdirectory of run.name as txt.

Author(s)

Marten Boetzer, Melle Sieswerda, Renee X. de Menezes <R.X.Menezes@lumc.nl>

See Also

SIM, tabulate.pvals, tabulate.top.dep.features

Examples

```
#first run example(assemble.data)
#and example(integrated.analysis)
table.indep <- tabulate.top.indep.features(input.regions="8q",
                                    adjust.method="BY",
            method="full",
            significance= 0.05,
            z.threshold=c(-1, 1),
            run.name="chr8q")
head(table.indep[["8q"]])
```

Index