Package 'PREDA'

November 12, 2026

Version 1.58.0

Title Position Related Data Analysis

Author Francesco Ferrari <francesco.ferrari@ifom.eu>

Maintainer Francesco Ferrari <francesco.ferrari@ifom.eu>

License GPL-2

Depends R (>= 2.9.0), Biobase, lokern (>= 1.0.9), multtest, stats, methods, annotate

Suggests quantsmooth, qvalue, limma, caTools, affy, PREDAsampledata

Enhances Rmpi, rsprng

Description Package for the position related analysis of quantitative functional genomics data.

Collate AllClasses.R AllGenerics.R analysesNames.R CleanNAAnnotationDataframe.R compareFunctionFromStatisticsForPREDA.R computeDatasetSignature.R DataForPREDA2dataframe.R DataForPREDA2GenomicAnnotationsForPREDA.R DataForPREDA2StatisticsForPREDA.R datasetSignatureFromFlags.R eset2GenomicAnnotations.R GE_computeStatistic.R GEhighWeakExpressionWorkflow.R genomePlot_improved.R genomePlot.R GenomicAnnotations2dataframe.R GenomicAnnotations2GenomicAnnotationsForPREDA.R GenomicAnnotations2reference_positions.R GenomicAnnotationsExtract.R GenomicAnnotationsFilter_neg.R GenomicAnnotationsFilter_pos.R GenomicAnnotationsForPREDA2dataframe.R GenomicAnnotationsForPREDA2GenomicAnnotations.R GenomicAnnotationsForPREDA2PREDAResults.R GenomicAnnotationsForPREDAFromfile.R GenomicAnnotationsForPREDAGetExpectedFlags.R GenomicAnnotationsFromdataframe.R GenomicAnnotationsFromfile.R GenomicAnnotationsFromLibrary.R GenomicAnnotationsSortAndCleanNA.R GenomicRegions2dataframe.R GenomicRegionsAnnotate.R GenomicRegionsChrNumber.R

GenomicRegionsComparison.R GenomicRegionsCreateRegionsIds.R
GenomicRegionsFilter_neg.R GenomicRegionsFilter_pos.R
GenomicRegionsFindOverlap.R GenomicRegionsFromdataframe.R
GenomicRegionsFromfile.R GenomicRegionsNumber.R
GenomicRegionsSpan.R GenomicRegionsTotalSpan.R
GEsimulationsWorkflow.R getExpectedSmoothFunction.R
getExpectedSmoothFunction_runmean.R getObservedSmoothFunction.R
getObservedSmoothFunction_runmean.R getStatisticByName.R
initialize-methods.R MergeStatisticAnnotations2DataForPREDA.R
PREDADataAndResults2dataframe.R PREDA_main_permRows.R
PREDA_main_permSamples.R PREDA_main.R
PREDA_multTestCorrection.R PREDA_quantsmoothStatPerm.R
PREDA_quantsmoothStat.R PREDAResults2dataframe.R
PREDAResults2GenomicRegions.R
PREDAResults2GenomicRegionsSingle.R
PREDAResults2PREDADataAndResults.R
PREDAResultsGetObservedFlags.R PREDA_smoothStatPerm.R
PREDA_smoothStat.R PREDA_splineStatPerm.R PREDA_splineStat.R
preprocessingGE.R RMAwithCDFfilter.R simulations.R
SODEGIR_GEstatistics.R SODEGIRpreprocessingGE.R
SortAnnotationDataframe.R StatisticsForPREDA2dataframe.R
StatisticsForPREDAFilterColumns_neg.R
StatisticsForPREDAFilterColumns_pos.R
StatisticsForPREDAFromdataframe.R statisticsForPREDAfromEset.R
StatisticsForPREDAFromfile.R

biocViews Software, CopyNumberVariation, GeneExpression, Genetics

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

git_branch RELEASE_3_23

git_last_commit cffa80d

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

analysesNames 4
computeDatasetSignature 5
DataForPREDA-class 6
DataForPREDA2dataframe 9
DataForPREDA2GenomicAnnotationsForPREDA 9
DataForPREDA2StatisticsForPREDA 10
DataForPREDAMedianCenter 10
eset2GenomicAnnotations 11
genomePlot 12
GenomicAnnotations-class 14
GenomicAnnotations2dataframe 15

GenomicAnnotations2GenomicAnnotationsForPREDA 16
GenomicAnnotations2reference_positions 17
GenomicAnnotationsExtract 18
GenomicAnnotationsFilter_neg 19
GenomicAnnotationsFilter_pos 19
GenomicAnnotationsForPREDA-class 20
GenomicAnnotationsForPREDA2dataframe 22
GenomicAnnotationsForPREDA2GenomicAnnotations 23
GenomicAnnotationsForPREDA2PREDAResults 23
GenomicAnnotationsForPREDAFromfile 24
GenomicAnnotationsFromdataframe 26
GenomicAnnotationsFromfile 27
GenomicAnnotationsFromLibrary 29
GenomicAnnotationsSortAndCleanNA 30
GenomicRegions-class 31
GenomicRegions2dataframe 33
GenomicRegionsAnnotate 34
GenomicRegionsChrNumber 35
GenomicRegionsComparison 36
GenomicRegionsCreateRegionsIds 37
GenomicRegionsFilter_neg 37
GenomicRegionsFilter_pos 38
GenomicRegionsFindOverlap 39
GenomicRegionsFromdataframe 40
GenomicRegionsFromfile 41
GenomicRegionsNumber 43
GenomicRegionsSpan 43
GenomicRegionsTotalSpan 43
getStatisticByName 44
MergeStatisticAnnotations2DataForPREDA 44
PREDADataAndResults-class 45
PREDADataAndResults2dataframe 47
PREDAResults-class 48
PREDAResults2dataframe 50
PREDAResults2GenomicRegions 51
PREDAResults2GenomicRegionsSingle 52
PREDAResults2PREDADataAndResults 53
PREDAResultsGetObservedFlags 53
PREDA_main 54
preprocessingGE 56
SODEGIRpreprocessingGE 58
SODEGIR_GEstatistics 61
StatisticsForPREDA-class 62
StatisticsForPREDA2dataframe 64
StatisticsForPREDAFilterColumns_neg 64
StatisticsForPREDAFilterColumns_pos 65
StatisticsForPREDAFromdataframe 65
statisticsForPREDAfromEset 67

StatisticsForPREDAFromfile 68

Index **70**

`analysesNames` *Get the names of the analyses in the from PREDA objects*

Description

Get the names of the analyses in the from StatisticsForPREDA objects, PREDAResults objects and objects from classes extending these classes.

Usage

```
analysesNames(.Object)
```

Arguments

`.Object`	an object of class StatisticsForPREDA, PREDAResults or any other class extending these classes

Value

Character vector of analysesNames

Author(s)

Francesco Ferrari

See Also

"StatisticsForPREDA", "PREDAResults"

Examples

```
require(PREDAsampledata)
data(SODEGIRGEanalysisResults)
analysesNames(SODEGIRGEanalysisResults)
```

computeDatasetSignature *Function to compute dataset signature for recurrent significant genomic regions*

Description

Function to compute dataset signature for recurrent significant genomic regions

Usage

```
# computeDatasetSignature(.Object, genomicRegionsList=genomicRegionsList,
# multTestCorrection="fdr", signature_qval_threshold=0.05,
# returnRegions=TRUE, use.referencePositions=TRUE)

computeDatasetSignature(.Object, ...)
```

Arguments

.Object	Object of class GenomicAnnotationsForPREDA
...	See below

genomicRegionsList: List of genomicRegions objects for which the recurrent overlapping regions will be evaluated

multTestCorrection: Multiple testing correction that will be adopted to correct the statistic p-values. Possible values are "fdr", for benjamini and Hochberg multiple testing correction and "qvalue" for p-values correction performed with qvalue package.

signature_qval_threshold: Threshold used to select significant regions resulting from the dataset signature statistic

returnRegions: Logical, if TRUE (default) the genomic regions constituting the daaset signature are returned, otherwise a PREDAresults object containing dataset signature statistics is returned.

use.referencePositions: Logical, if TRUE the input reference positions used for PREDA analysis wil be used to identify significant genomic regions boundaries as well.

Details

The function adopts a binomial test to identify significant recurrence of genomic regions across multiple dataset sampels.

Value

A GenomicRegions object (if returnRegions = TRUE) or a PREDAresults object containing dataset signature statistics (if returnRegions = FALSE)

Author(s)

Francesco Ferrari

See Also

GenomicRegions, PREDAResults

Examples

```
## Not run:

require(PREDAsampledata)
data(SODEGIRCNanalysisResults)
data(GEDataForPREDA)

SODEGIR_CN_GAIN<-PREDAResults2GenomicRegions(
SODEGIRCNanalysisResults, qval.threshold=0.01,
smoothStatistic.tail="upper", smoothStatistic.threshold=0.1)


CNgain_signature<-computeDatasetSignature(GEDataForPREDA,
genomicRegionsList=SODEGIR_CN_GAIN)


## End(Not run)
```

DataForPREDA-class *Class "DataForPREDA" is used to manage all of the data required as input for PREDA analysis*

Description

This class is used to manage all of the data required as input for PREDA analysis: it is usually created by merging a GenomicAnnotationsForPREDA and a StatisticsForPREDA classes

Objects from the Class

Objects can be created by calls of the form `new("DataForPREDA", ids, chr, start, end, strand, chromosomesNumbers, chromosomesLabels, position, optionalAnnotations, optionalAnnotationsHeaders, statistic, analysesNames, testedTail)`.

Slots

`position`: Object of class `"integer"` ~~

`ids`: Object of class `"character"` a character vector of unique identifiers for the genomic features under investigation

`chr`: Object of class `"integer"` a numeric vector representing the chromosome where each ids is mapped. Please note that chromosome usually not represented with a number must will be comverted to a number as well. e.g. for Human, chromsomomees X and Y will be converted to chromsomes 23 and 24 respectively. User defined options will allow this conversion during GenomicAnnotations objects initialization.

`start`: Object of class `"integer"` a numeric vector of start genomic position for each genomic feature under investigation (i.e. gene, transcript, SNP or other elements).

`end`: Object of class `"integer"` a numeric vector of end genomic position for each genomic feature under investigation (i.e. gene, transcript, SNP or other elements).

`strand`: Object of class `"numeric"` a numeric vector of strand genomic position for each genomic feature under investigation: value 1 is used for "plus" (forward) strand and value -1 for "minus" (reverse) strand. User defined options will allow the conversion to this format during GenomicAnnotations objects initialization.

`chromosomesNumbers`: Object of class `"numeric"` a numeric vector containing the list of chromosomes for which genomic annotations are provided in the GenomicAnnotations object. Each chromosome is represented just once in increasing order. Please note that chromosome usually not represented with a number must will be comverted to a number as well. e.g. for Human, chromsomomees X and Y will be converted to chromsomes 23 and 24 respectively.

`chromosomesLabels`: Object of class `"character"` a character vector containing the list of chromosomes for which genomic annotations are provided in the GenomicAnnotations object. Each chromosome is represented just once in the same order as reported in chromosomesNumbers slot. This slot is actually used just to provide a label for each associated chromosome number, in case that some non numeric chromsome is used (e.g. to preserve the correspondence between chr 23 and the actual chr X in Human)

`optionalAnnotations`: Object of class `"matrix"` optional annotations associated to the genomic features can be managed along with genomic positions annotations. E.g. GeneSymbol or EntrezGene ids can be associated to gene realted GenomicAnnotaitons objects. These additional annotations are not mandatory (the default value for this slot is NULL) The additional annotations must be provided as a matrix of character, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "optionalAnnotationsHeaders" slot.

`optionalAnnotationsHeaders`: Object of class `"character"` character vector containing the names associated to optional annotations. Please avoid using spaces in annotations names.

`statistic`: Object of class `"matrix"` a numeric matrix containing gene-centered statistics (or statistics on genomic data centered on other genomic features under investigation). The statistics must be provided as a matrix of numeric values, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "analysesNames" slot.

`analysesNames`: Object of class `"character"` a character vector of unique names associated to each column of statistic matrix. This is just a name that will be used to identify each analysis.

`testedTail`: Object of class `"character"` a character describing what tail of the statistic distribution will be analyzed during PREDA analysis. Possible values are "upper", "lower" or "both". Anyway we strongly recommend using PREDA analysis only for statistics on genomic data with a symmetric distribution around zero.

Extends

Class `"GenomicAnnotationsForPREDA"`, directly. Class `"StatisticsForPREDA"`, directly. Class `"GenomicAnnotations"`, by class "GenomicAnnotationsForPREDA", distance 2.

Methods

DataForPREDA2dataframe `signature(.Object = "DataForPREDA")`: extract data and annotations as a dataframe with probeids as rownames

DataForPREDA2GenomicAnnotationsForPREDA `signature(.Object = "DataForPREDA")`: extract a GenomicAnnotationsForPREDA object from a data DataForPREDA object

DataForPREDA2StatisticsForPREDA `signature(.Object = "DataForPREDA")`: extract a StatisticsForPREDA object from a data DataForPREDA object

GenomicAnnotationsFilter_neg `signature(.Object = "DataForPREDA")`: filter annotations to remove selected chromosomes

GenomicAnnotationsFilter_pos `signature(.Object = "DataForPREDA")`: filter annotations to keep selected chromosomes

GenomicAnnotationsSortAndCleanNA `signature(.Object = "DataForPREDA")`: sort annotations according to selected chromosomes and to remove genes containing any NA annotation field

initialize `signature(.Object = "DataForPREDA")`: initialize method for DataForPREDA objects

StatisticsForPREDAFilterColumns_neg `signature(.Object = "DataForPREDA")`: filter statistics to remove selected analyses

StatisticsForPREDAFilterColumns_pos `signature(.Object = "DataForPREDA")`: filter statistics to keep selected analyses

Note

This class is better described in the package vignette

Author(s)

Francesco Ferrari

See Also

"GenomicAnnotations", "GenomicAnnotationsForPREDA", "StatisticsForPREDA", DataForPREDA2dataframe,DataFo
GenomicAnnotationsFilter_neg,GenomicAnnotationsFilter_pos,GenomicAnnotationsSortAndCleanNA,
StatisticsForPREDAFilterColumns_neg,StatisticsForPREDAFilterColumns_pos

Examples

```
showClass("DataForPREDA")
```

`DataForPREDA2dataframe` *extract data and annotations as a dataframe*

Description

extract data and annotations as a dataframe with probeids as rownames

Usage

```
DataForPREDA2dataframe(.Object)
```

Arguments

`.Object`	An object of class DataForPREDA

Details

extract data and annotations as a dataframe with probeids as rownames

Value

a dataframe with probeids as rownames

`DataForPREDA2GenomicAnnotationsForPREDA` *extract a GenomicAnnotationsForPREDA object from a data DataForPREDA object*

Description

extract a GenomicAnnotationsForPREDA object from a data DataForPREDA object

Usage

```
DataForPREDA2GenomicAnnotationsForPREDA(.Object)
```

Arguments

`.Object`	an object of class DataForPREDA

Details

extract a GenomicAnnotationsForPREDA object from a data DataForPREDA object

Value

a GenomicAnnotationsForPREDA object

`DataForPREDA2StatisticsForPREDA` *extract a StatisticsForPREDA object from a data DataForPREDA object*

Description

extract a StatisticsForPREDA object from a data DataForPREDA object

Usage

```
DataForPREDA2StatisticsForPREDA(.Object)
```

Arguments

`.Object`	a data DataForPREDA object

Details

extract a StatisticsForPREDA object from a data DataForPREDA object

Value

a StatisticsForPREDA object

`DataForPREDAMedianCenter` *Function to scale median value of DataForPREDA statistics to zero*

Description

Function to scale median value of DataForPREDA statistics to zero

Usage

```
DataForPREDAMedianCenter(.Object, ...)
```

Arguments

`.Object`	a DataForPREDA object
`...`	

Details

Scale median value of DataForPREDA statistics to zero

Value

a DataForPREDA object

`eset2GenomicAnnotations` *Function building a GenomicAnnotations object on an ExpressionSet object*

Description

Function building a GenomicAnnotations object on an ExpressionSet object

Usage

```
# eset2GenomicAnnotations(.Object, retain.chrs,
# optionalAnnotations)

eset2GenomicAnnotations(.Object, ...)
```

Arguments

`.Object`	ExpressionSet object. The associated annotation library will be used to build a GenomicAnnotations object.
`...`	See below

retain.chrs: Numeric vector, containing the list of chromosomes selected for the output GenomicAnnotations object. E.g. set retain.chrs=1:22 to limit the GenomicAnnotations object to chromosomes from 1 to 22. This might be ueseful to limit GenomiAnnotations objects to autosomic chromosomes.

optionalAnnotations: Character vector to select additional annotations fields to be included into the GenomicAnnotations object.

Value

An object of class "GenomicAnnotations"

Author(s)

Francesco Ferrari

See Also

"GenomicAnnotations"

Examples

```
  ## Not run:
require("PREDAsampledata")
data(ExpressionSetRCC)

GEGenomicAnnotations<-eset2GenomicAnnotations(ExpressionSetRCC,
retain.chrs=1:22)


## End(Not run)
```

---

genomePlot *draw a genome plot*

Description

draw a genome plot with user defined genomic regions

Usage

```
# genomePlot(.Object, genomicRegions=NULL, draw.blocks=TRUE,
# parallel.plot=TRUE, grouping=NULL, custom.labels=NULL,
#  scale.positions=NULL, qval.threshold=0.05,
# use.referencePositions=FALSE, smoothStatistic.tail=NULL,
# smoothStatistic.threshold=NULL, region.colors=NULL,
# limitChrs=NULL)

genomePlot(.Object, ...)
```

Arguments

.Object	Object of class GenomicAnnotationsForPREDA, or any other class exteinding this one.
...	See below

genomicRegions: A list of GenomicRegions object containing the genomic regions to be highlighted in the plot.

draw.blocks: If TRUE genomic regions are plotted as blocks. Otherwise they are plotted as coloured ticks. Currently only draw.blocks=TRUE is implemented.

parallel.plot: Logical, if TRUE multiple copies of each chrosomosome are drawn.
In particular a number of copies equal to lnegth(grouping), if grouping is not null, or a number of copies equal to the number of GenomicRegions objects provided as input.

grouping: Vector specifying how input GenomicRegions objects will be grouped on chromosomes.

custom.labels: A character to specify user defined labels for vertical axis

scale.positions: Parameter to set the scale for chromosomal positions (horizontal axis). Possible values are "Mb" or "Kb"

qval.threshold: If genomicRegions is NULL, and a PREDAResults or PREDADataAndResults is provided as input, the function PREDAResults2GenomcRegions is applied with this parameter to extract significant GenomicRegions.

use.referencePositions: If genomicRegions is NULL, and a PREDAResults or PREDADataAndResults is provided as input, the function PREDAResults2GenomcRegions is applied with this parameter to extract significant GenomicRegions.

smoothStatistic.tail: If genomicRegions is NULL, and a PREDAResults or PREDADataAndResults is provided as input, the function PREDAResults2GenomcRegions is applied with this parameter to extract significant GenomicRegions.

smoothStatistic.threshold: If genomicRegions is NULL, and a PREDAResults or PREDADataAndResults is provided as input, the function PREDAResults2GenomcRegions is applied with this parameter to extract significant GenomicRegions.

region.colors: Character vector specifyin the list of colors to be used for drawing each set of GenomicRegions. Mut be of length equal to the number of GenomicRegions objects provided as input.

limitChrs: Numeric vector, that can be used to limit the plot to a subset of chromosomes.

Details

See also the PREDA tutorial vignette for more details and sample usage

Value

A plot of the genome with significant GenomicRegions

Author(s)

Francesco Ferrari

See Also

PREDAResults2GenomicRegions, PREDAResults, PREDADataAndResults, GenomicAnnotationsForPREDA

Examples

```
## See PREDA tutorial vignette for some examples
```

`GenomicAnnotations-class` *Class "GenomicAnnotations" to manage information about genomic features*

Description

This class is used to manage information about genomic features under investigation: i.e. genomic genes, SNP or others, with particular focus on the genomic coordinates of each of them. Other additional annotations associated to each element can be stored in a GenomicAnnotations object in the optionalAnnotations slots

Objects from the Class

Objects can be created by calls of the form `new("GenomicAnnotations",ids, chr, start, end, strand, chromosomesNumbers, chromosomesLabels,optionalAnnotations, optionalAnnotationsHeaders)`.

Slots

`ids`: Object of class `"character"` ~~

`chr`: Object of class `"integer"` ~~

`start`: Object of class `"integer"` ~~

`end`: Object of class `"integer"` ~~

`strand`: Object of class `"numeric"` ~~

`chromosomesNumbers`: Object of class `"numeric"` ~~

`chromosomesLabels`: Object of class `"character"` ~~

`optionalAnnotations`: Object of class `"matrix"` ~~

`optionalAnnotationsHeaders`: Object of class `"character"` ~~

Methods

GenomicAnnotations2dataframe `signature(.Object = "GenomicAnnotations")`: extracts annotations as a dataframe with probeids as rownames

GenomicAnnotations2GenomicAnnotationsForPREDA `signature(.Object = "GenomicAnnotations")`: generate a new GenomicAnnotationsForPREDA object from a GenomicAnnotations object

GenomicAnnotations2reference_positions `signature(.Object = "GenomicAnnotations")`: extract from the GenomicAnnotations object a vector containing a vector with reference positions

GenomicAnnotationsExtract `signature(.Object = "GenomicAnnotations")`: extract optional annotations for a specific region

GenomicAnnotationsFilter_neg `signature(.Object = "GenomicAnnotations")`: filter annotations to remove selected chromosomes

GenomicAnnotationsFilter_pos `signature(.Object = "GenomicAnnotations")`: filter annotations to keep selected chromosomes

GenomicAnnotationsSortAndCleanNA `signature(.Object = "GenomicAnnotations")`: sort annotations according to selected chromosomes and to remove genes containing any NA annotation field

GenomicRegionsAnnotate `signature(.Object1 = "GenomicRegions", .Object2 = "GenomicAnnotations")`: extract annotations from a GenomicAnnotations object for a set of regions specified as a GenomicRegions object

initialize `signature(.Object = "GenomicAnnotations")`: initialize method for GenomicAnnotations objects

Note

This class is better described in the package vignette

Author(s)

Francesco Ferrari

See Also

GenomicAnnotations2dataframe, GenomicAnnotations2GenomicAnnotationsForPREDA, GenomicAnnotations2refere
GenomicAnnotationsFilter_neg,GenomicAnnotationsFilter_pos, GenomicAnnotationsSortAndCleanNA,GenomicRe

Examples

```
showClass("GenomicAnnotations")
```

GenomicAnnotations2dataframe

extracts annotations as a dataframe

Description

extracts annotations as a dataframe with probeids as rownames

Usage

```
GenomicAnnotations2dataframe(.Object)
```

Arguments

`.Object`	A GenomicAnnotations object

Details

extract annotations as a dataframe with probeids as rownames

Value

a dataframe with probeids as rownames

`GenomicAnnotations2GenomicAnnotationsForPREDA` *generate a GenomicAnnotationsForPREDA object from a GenomicAnnotations object*

Description

generate a new GenomicAnnotationsForPREDA object from a GenomicAnnotations object

Usage

```
# GenomicAnnotations2GenomicAnnotationsForPREDA(.Object,
# positions=NULL, reference_position_type=NULL)

GenomicAnnotations2GenomicAnnotationsForPREDA(.Object,
... )
```

Arguments

`.Object`	An object of class GenomicAnnotations
`...`	See below

positions: Vector to specify reference positions for GenomicAnnotationsForPREDA object if not specified with reference_position_type parameter

reference_position_type: Specify which genomic coordinate must be used as reference position for PREDA analysis. Possible values are "start", "end", "median", "strand.start" or "strand.end".
"strand.start" is strand specific start: i.e. start on positive strand but end on negative strand. "strand.end" is strand specific end.

Value

A GenomicAnnotationsForPREDA object

Author(s)

Francesco Ferrari

See Also

`GenomicAnnotationsForPREDA`

Examples

```
 ## Not run:

GEGenomicAnnotations<-GenomicAnnotationsFromLibrary(annotLibrary
= "org.Hs.eg.db", retain.chrs=1:22)

  GEGenomicAnnotationsForPREDA<-
  GenomicAnnotations2GenomicAnnotationsForPREDA(
  GEGenomicAnnotations, reference_position_type="median")


## End(Not run)
```

`GenomicAnnotations2reference_positions`
extract reference positions from the GenomicAnnotations

Description

extract from the GenomicAnnotations object a vector containing a vector with reference positions

Usage

```
# GenomicAnnotations2reference_positions(.Object,
# reference_position_type=c("start", "end", "median", "strand.start", "strand.end"),
# withnames=TRUE)

GenomicAnnotations2reference_positions(.Object, ...)
```

Arguments

`.Object`	Object of class GenomicAnnotations
`...`	See below

reference_position_type: Specify which genomic coordinate must be used as reference position for PREDA analysis. Possible values are "start", "end", "median", "strand.start" or "strand.end".
"strand.start" is strand specific start: i.e. start on positive strand but end on negative strand. "strand.end" is strand specific end.

withnames: Logical, if TRUE the "ids" slot content is used as names for the output vector

Value

A numeric vector with the selected reference positions.

GenomicAnnotationsExtract *extract optional annotations for a specific region*

Description

extract optional annotations for a specific region

Usage

```
# GenomicAnnotationsExtract(.Object, chr, start, end,
# AnnotationsHeader=NULL, sep.character="; ",
# complete.inclusion=FALSE, skipSorting=FALSE,
# annotationAsRange=FALSE, getJustFeaturesNumber=FALSE)

GenomicAnnotationsExtract(.Object, ...)
```

Arguments

.Object	An object of class GenomicAnnotations
...	See below

chr: Coordinate for the selected genomic region

start: Coordinate for the selected genomic region

end: Coordinate for the selected genomic region

AnnotationsHeader: Character or numeric vector to select the annotations columns to be considered

sep.character: Character used to separate annotated features in the ouptput

complete.inclusion: Logical, if TRUE only annotated features completely included in the region are reported. If FALSE (default), every overlapping the feature is considered.

skipSorting: Logical, if TRUE, annotation sorting is skipped before processing output (to save computational time, e.g. in a long loop)

annotationAsRange: If TRUE, then only the first and last annotated element in the region are reported

getJustFeaturesNumber: Logical: if TRUE, just the number of annotated features in the region is returned

Details

Extract annotations associated to a specific genomic region from a GenomiAnnotations object. Only annotations from the specified columns are returned.

Value

A character vector is returned

See Also

"GenomicAnnotations"

GenomicAnnotationsFilter_neg

filter annotations to remove selected chromosomes

Description

filter annotations to remove selected chromosomes

Usage

```
# GenomicAnnotationsFilter_neg(.Object, chrToRemove, chrAsLabels=FALSE)

GenomicAnnotationsFilter_neg(.Object, ...)
```

Arguments

.Object	An object of class GenomicAnnotations or classes inheriting from GenomicAnnotations
...	See below

chrToRemove: List of chromosomes to be removed from the annotations object.

chrAsLabels: Logical, TRUE if chromosomes are listed as their character labels, instead of using the numeric indexes

---

GenomicAnnotationsFilter_pos

filter annotations to keep selected chromosomes

Description

filter annotations to keep selected chromosomes

Usage

```
# GenomicAnnotationsFilter_pos(.Object, chrToRetain, chrAsLabels=FALSE)

GenomicAnnotationsFilter_pos(.Object, ...)
```

Arguments

`.Object`	An object of class GenomicAnnotations or classes inheriting from GenomicAnnotations
`...`	See below
	chrToRetain: List of chromosomes to be maintained after removing the annotations for all the other chromosomes.
	chrAsLabels: Logical, TRUE if chromosomes are listed as their character labels, instead of using the numeric indexes

---

`GenomicAnnotationsForPREDA-class`

Class "GenomicAnnotationsForPREDA" GenomicAnnotations class with additional slot specifying the reference position for PREDA analysis

---

Description

This class is equivalent to the GenomicAnnotations class but includes an additional slot specifying the reference position that will be used for PREDA smoothing of data: this is included in the "position" slot. An unique reference position is required for PREDA analysis because this position is used for smoothing data along chromosomal coordinates. This reference position usaually is the start, the end, or the median posizion of each considered genomic feature, nevertheless other user defined positions could be used as well.

Objects from the Class

Objects can be created by calls of the form `new("GenomicAnnotationsForPREDA", ids, chr, start, end, strand, chromosomesNumbers, chromosomesLabels, position, optionalAnnotations, optionalAnnotationsHeaders)`.

Slots

`position`**:** Object of class `"integer"` a numeric vector of reference genomic positions that will be associated and used for each genomic feature under investigation for smoothing data during PREDA analysis.

`ids`**:** Object of class `"character"` a character vector of unique identifiers for the genomic features under investigation

`chr`**:** Object of class `"integer"` a numeric vector representing the chromosome where each ids is mapped. Please note that chromosome usually not represented with a number must will be comverted to a number as well. e.g. for Human, chromsomomees X and Y will be converted to chromsomes 23 and 24 respectively. User defined options will allow this conversion during GenomicAnnotations objects initialization.

`start`**:** Object of class `"integer"` a numeric vector of start genomic position for each genomic feature under investigation (i.e. gene, transcript, SNP or other elements).

`end`: Object of class `"integer"` a numeric vector of end genomic position for each genomic feature under investigation (i.e. gene, transcript, SNP or other elements).

`strand`: Object of class `"numeric"` a numeric vector of strand genomic position for each genomic feature under investigation: value 1 is used for "plus" (forward) strand and value -1 for "minus" (reverse) strand. User defined options will allow the conversion to this format during GenomicAnnotations objects initialization.

`chromosomesNumbers`: Object of class `"numeric"` a numeric vector containing the list of chromosomes for which genomic annotations are provided in the GenomicAnnotations object. Each chromosome is represented just once in increasing order. Please note that chromosome usually not represented with a number must will be comverted to a number as well. e.g. for Human, chromsomomees X and Y will be converted to chromsomes 23 and 24 respectively.

`chromosomesLabels`: Object of class `"character"` a character vector containing the list of chromosomes for which genomic annotations are provided in the GenomicAnnotations object. Each chromosome is represented just once in the same order as reported in chromosomesNumbers slot. This slot is actually used just to provide a label for each associated chromosome number, in case that some non numeric chromsome is used (e.g. to preserve the correspondence between chr 23 and the actual chr X in Human)

`optionalAnnotations`: Object of class `"matrix"` optional annotations associated to the genomic features can be managed along with genomic positions annotations. E.g. GeneSymbol or EntrezGene ids can be associated to gene realted GenomicAnnotaitons objects. These additional annotations are not mandatory (the default value for this slot is NULL) The additional annotations must be provided as a matrix of character, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "optionalAnnotationsHeaders" slot.

`optionalAnnotationsHeaders`: Object of class `"character"` character vector containing the names associated to optional annotations. Please avoid using spaces in annotations names.

Extends

Class `"GenomicAnnotations"`, directly.

Methods

genomePlot `signature(.Object = "GenomicAnnotationsForPREDA")`: draw a genome plot

GenomicAnnotations2dataframe `signature(.Object = "GenomicAnnotationsForPREDA")`: extract annotations as a dataframe with probeids as rownames

GenomicAnnotationsFilter_neg `signature(.Object = "GenomicAnnotationsForPREDA")`: filter annotations to remove selected chromosomes

GenomicAnnotationsFilter_pos `signature(.Object = "GenomicAnnotationsForPREDA")`: filter annotations to keep selected chromosomes

GenomicAnnotationsForPREDA2dataframe `signature(.Object = "GenomicAnnotationsForPREDA")`: extract annotations as a dataframe with probeids as rownames

GenomicAnnotationsForPREDA2GenomicAnnotations `signature(.Object = "GenomicAnnotationsForPREDA")`: extract the GenomicAnnotations object from the GenomicAnnotationsForPREDA object

GenomicAnnotationsForPREDA2PREDAResults `signature(.Object = "GenomicAnnotationsForPREDA")`: add PREDA results information to genomic annotatations creating a PREDAResults object

GenomicAnnotationsSortAndCleanNA `signature(.Object = "GenomicAnnotationsForPREDA")`: sort annotations according to selected chromosomes and to remove genes containing any NA annotation field

initialize `signature(.Object = "GenomicAnnotationsForPREDA")`: initialize method for GenomicAnnotationsForPREDA objects

Note

This class is better described in the package vignette

Author(s)

Francesco Ferrari

See Also

"GenomicAnnotations", GenomicAnnotationsSortAndCleanNA, GenomicAnnotationsForPREDA2PREDAResults,Genomi
GenomicAnnotationsForPREDA2dataframe,GenomicAnnotationsFilter_pos, GenomicAnnotationsFilter_neg,Genom

Examples

```
showClass("GenomicAnnotationsForPREDA")
```

GenomicAnnotationsForPREDA2dataframe

extract annotations as a dataframe

Description

extract annotations as a dataframe with probeids as rownames

Usage

```
GenomicAnnotationsForPREDA2dataframe(.Object)
```

Arguments

`.Object`	an object of class GenomicAnnotationsForPREDA

Details

extract annotations from an object of class GenomicAnnotationsForPREDA as a dataframe with probeids as rownames

Value

a dataframe with probeids as rownames

GenomicAnnotationsForPREDA2GenomicAnnotations
extract the GenomicAnnotations object from the GenomicAnnotationsForPREDA object

Description

extract the GenomicAnnotations object from the GenomicAnnotationsForPREDA object

Usage

```
GenomicAnnotationsForPREDA2GenomicAnnotations(.Object)
```

Arguments

.Object	an object of class GenomicAnnotationsForPREDA

GenomicAnnotationsForPREDA2PREDAResults
add PREDA results information to genomic annotatations creating a PREDAResults object

Description

add PREDA results information to genomic annotatations creating a PREDAResults object

Usage

```
# GenomicAnnotationsForPREDA2PREDAResults(.Object, analysesNames, testedTail, smoothStatistic, pvalue

GenomicAnnotationsForPREDA2PREDAResults(.Object, ...)
```

Arguments

.Object	An object of class GenomicAnnotationsForPREDA
...	See below
	analysesNames: analysesNames as in PREDAResults object
	testedTail: testedTail as in PREDAResults object
	smoothStatistic: smoothStatistic as in PREDAResults object
	pvalue: pvalue as in PREDAResults object
	qvalue: qvalue as in PREDAResults object

GenomicAnnotationsForPREDAFromfile

Function to create a GenomicAnnotationsForPREDA object from a txt file

Description

Function to create a GenomicAnnotationsForPREDA object from a txt file

Usage

```
GenomicAnnotationsForPREDAFromfile(file, ids_column, chr_column,
start_column, end_column, strand_column, chromosomesNumbers =
NULL, chromosomesLabels = NULL, chromosomesLabelsInput = NULL,
MinusStrandString = "-", PlusStrandString = "+",
optionalAnnotationsColumns = NULL, reference_position_type =
"median", ...)
```

Arguments

file	Path to the input txt file containing genomic annotations
ids_column	Specify the column from the input txt file with gene (or other genomic features) ids. Can be specified using column index (numeric) or column name (character).
chr_column	Specify the column from the input txt file with chromosome annotations fields for each ids. Can be specified using column index (numeric) or column name (character).
start_column	Specify the column from the input txt file with genomic start position for each genomic element. Can be specified using column index (numeric) or column name (character).
end_column	Specify the column from the input txt file with genomic end position for each genomic element. Can be specified using column index (numeric) or column name (character).
strand_column	Specify the column from the input txt file with genomic strand mapping for each genomic element. Can be specified using column index (numeric) or column name (character).
chromosomesNumbers	Numeric vector to specify the list of numeric values to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)
chromosomesLabels	Character vector to specify the list of character labels to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)

`chromosomesLabelsInput`
: Character vector to specify the list of character labels associated to each chromosome in the input file. Particularly useful when non numeric character strings are associated to eacforh chromosome in the input file: e.g. "chr3" for chromosome "3".

`MinusStrandString`
: Character string used to identify minus strand in the input text file

`PlusStrandString`
: Character string used to identify plus strand in the input text file

`optionalAnnotationsColumns`
: Character vector of columns headers or numeric vector of columns indices to specify columns of the input file containing additional annotation fields

`reference_position_type`
: Character string to specify which genomic coordinate must be used as reference position for PREDA analysis. See also "GenomicAnnotations2GenomicAnnotationsForPREDA"

`...`
: any other parameter for read.table function that could be useful for parsing the input file, such as "sep", "quote", "header", "na.strings" and other parameters.

Value

An object of class "GenomicAnnotationsForPREDA"

Author(s)

Francesco Ferrari

See Also

"GenomicAnnotationsForPREDA"

Examples

```
  ## Not run:

data(PREDAsampledata)
 CNdataPath <- system.file("sampledata", "CopyNumber", package =
"PREDAsampledata")
CNannotationFile <- file.path(CNdataPath , "SNPAnnot100k.csv")

CNGenomicsAnnotations<-GenomicAnnotationsForPREDAFromfile(
  file=CNannotationFile,
  ids_column=1,
  chr_column="Chromosome",
  start_column=4,
  end_column=4,
  strand_column="Strand",
  chromosomesLabelsInput=1:22,
  MinusStrandString="-", PlusStrandString="+",
  optionalAnnotationsColumns=c("Cytoband", "Entrez_gene"),
  header=TRUE, sep=",", quote="\"", na.strings = c("NA", "",
```

```
  "---"))


## End(Not run)
```

GenomicAnnotationsFromdataframe
Function to create a GenomiAnnotations object from a dataframe

---

Description

Function to create a GenomiAnnotations object from a dataframe

Usage

```
GenomicAnnotationsFromdataframe(GenomicAnnotations_dataframe, ids_column, chr_column,
start_column, end_column, strand_column, chromosomesNumbers =
NULL, chromosomesLabels = NULL, chromosomesLabelsInput = NULL,
MinusStrandString = "-", PlusStrandString =
"+", optionalAnnotationsColumns = NULL)
```

Arguments

GenomicAnnotations_dataframe	Dataframe object contanining genomic annotations.
ids_column	Specify the column from the input txt file with gene (or other genomic features) ids. Can be specified using column index (numeric) or column name (character).
chr_column	Specify the column from the input txt file with chromosome annotations fields for each ids. Can be specified using column index (numeric) or column name (character).
start_column	Specify the column from the input txt file with genomic start position for each genomic element. Can be specified using column index (numeric) or column name (character).
end_column	Specify the column from the input txt file with genomic end position for each genomic element. Can be specified using column index (numeric) or column name (character).
strand_column	Specify the column from the input txt file with genomic strand mapping for each genomic element. Can be specified using column index (numeric) or column name (character).
chromosomesNumbers	Numeric vector to specify the list of numeric values to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)

`chromosomesLabels`
: Character vector to specify the list of character labels to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)

`chromosomesLabelsInput`
: Character vector to specify the list of character labels associated to each chromosome in the input file. Particularly useful when non numeric character strings are associated to eacforh chromosome in the input file: e.g. "chr3" for chromosome "3".

`MinusStrandString`
: Character string used to identify minus strand in the input text file

`PlusStrandString`
: Character string used to identify plus strand in the input text file

`optionalAnnotationsColumns`
: Character vector of columns headers or numeric vector of columns indices to specify columns of the input file containing additional annotation fields

Value

An object of class "GenomicAnnotations"

Author(s)

Francesco Ferrari

See Also

"GenomicAnnotations"

`GenomicAnnotationsFromfile` *Function to create a GenomiAnnotations object from a text file*

Description

Function to create a GenomiAnnotations object from a text file

Usage

```
GenomicAnnotationsFromfile(file, ids_column, chr_column,
start_column, end_column, strand_column, chromosomesNumbers =
NULL, chromosomesLabels = NULL, chromosomesLabelsInput = NULL,
MinusStrandString = "-", PlusStrandString =
"+", optionalAnnotationsColumns = NULL, ...)
```

Arguments

file	Path to the input txt file containing genomic annotations
ids_column	Specify the column from the input txt file with gene (or other genomic features) ids. Can be specified using column index (numeric) or column name (character).
chr_column	Specify the column from the input txt file with chromosome annotations fields for each ids. Can be specified using column index (numeric) or column name (character).
start_column	Specify the column from the input txt file with genomic start position for each genomic element. Can be specified using column index (numeric) or column name (character).
end_column	Specify the column from the input txt file with genomic end position for each genomic element. Can be specified using column index (numeric) or column name (character).
strand_column	Specify the column from the input txt file with genomic strand mapping for each genomic element. Can be specified using column index (numeric) or column name (character).
chromosomesNumbers	Numeric vector to specify the list of numeric values to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)
chromosomesLabels	Character vector to specify the list of character labels to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)
chromosomesLabelsInput	Character vector to specify the list of character labels associated to each chromosome in the input file. Particularly useful when non numeric character strings are associated to eacforh chromosome in the input file: e.g. "chr3" for chromosome "3".
MinusStrandString	Character string used to identify minus strand in the input text file
PlusStrandString	Character string used to identify plus strand in the input text file
optionalAnnotationsColumns	Character vector of columns headers or numeric vector of columns indices to specify columns of the input file containing additional annotation fields
...	any other parameter for read.table function that could be useful for parsing the input file, such as "sep", "quote", "header", "na.strings" and other parameters.

Value

An object of class "GenomicAnnotations"

Author(s)

Francesco Ferrari

See Also

"GenomicAnnotations"

Examples

```
  ## Not run:

data(PREDAsampledata)
 CNdataPath <- system.file("sampledata", "CopyNumber", package =
"PREDAsampledata")
CNannotationFile <- file.path(CNdataPath , "SNPAnnot100k.csv")

CNGenomicsAnnotations<-GenomicAnnotationsForPREDAFromfile(
  file=CNannotationFile,
  ids_column=1,
  chr_column="Chromosome",
  start_column=4,
  end_column=4,
  strand_column="Strand",
  chromosomesLabelsInput=1:22,
  MinusStrandString="-", PlusStrandString="+",
  optionalAnnotationsColumns=c("Cytoband", "Entrez_gene"),
  header=TRUE, sep=",", quote="\"", na.strings = c("NA", "",
  "---"))



## End(Not run)
```

GenomicAnnotationsFromLibrary

Function extracting a GenomicAnnotations object from a Bioconductor annotation library

---

Description

Function extracting a GenomicAnnotations object from a Bioconductor annotation library

Usage

```
GenomicAnnotationsFromLibrary(annotLibrary, probeIDs = NULL,
retain.chrs = NULL, optionalAnnotations = NULL)
```

Arguments

annotLibrary	Character string containing the name of the annotations library to be used for building the GenomicAnnotations object
probeIDs	Optional: list of reference id from the selected annotLibrary to be used for building the GenomicAnnotations object

`retain.chrs` Numeric vector, containing the list of chromosomes selected for the output GenomicAnnotations object. E.g. set retain.chrs=1:22 to limit the GenomicAnnotations object to chromosomes from 1 to 22. This might be ueseful to limit GenomiAnnotations objects to autosomic chromosomes.

`optionalAnnotations` Character vector to select additional annotations fields to be included into the GenomicAnnotations object.

Value

An object of class "GenomicAnnotations"

Author(s)

Francesco Ferrari

See Also

"GenomicAnnotations"

Examples

```
## Not run:

GEGenomicAnnotations<-GenomicAnnotationsFromLibrary(annotLibrary=
"org.Hs.eg.db", retain.chrs=1:22)

# with optional annotations Genesymbols and EntrezGeneIDs
GEGenomicAnnotations<-GenomicAnnotationsFromLibrary(annotLibrary=
"hgu133plus2.db", retain.chrs=1:22,
optionalAnnotations=c("SYMBOL", "ENTREZID"))



## End(Not run)
```

GenomicAnnotationsSortAndCleanNA

sort annotations according to selected chromosomes and to remove genes containing any NA annotation field

---

Description

sort annotations according to selected chromosomes and to remove genes containing any NA annotation field

Usage

```
# GenomicAnnotationsSortAndCleanNA(.Object, sorting_position_column="start")

GenomicAnnotationsSortAndCleanNA(.Object, ...)
```

Arguments

.Object	An object of class GenomicAnnotations or any object inheriting from GenomicAnnotations
...	See below
	sorting_position_column: Annotations slot used to sort data within each chromosome. Possilbe values include "start", "end" or "position" (the last one for GenomicAnnotationsForPREDA objects)

GenomicRegions-class *Class "GenomicRegions" is used to manage information about genomic regions*

Description

This class is used to manage genomic regions information that can be derived from PREDA analysis results or from other sources:e.g. relevant genomic regions from literature reports can be imported into a GenomicRegions object and compared with PREDA analysis results

Objects from the Class

Objects can be created by calls of the form `new("GenomicRegions", chr, start, end, chromosomesNumbers, chromosomesLabels, optionalAnnotations, optionalAnnotationsHeaders, ids)`.

Slots

chr: Object of class `"integer"` a numeric vector representing the chromosome where each genomic region is located. Please note that chromosome usually not represented with a number must will be comverted to a number as well. e.g. for Human, chromsomomes X and Y will be converted to chromsomes 23 and 24 respectively. User defined options will allow this conversion during GenomicAnnotations objects initialization.

start: Object of class `"integer"` a numeric vector of start genomic position for each genomic region. This vector must have the same length of "chr" slot.

end: Object of class `"integer"` a numeric vector of end genomic position for each genomic region. This vector must have the same length of "chr" slot.

chromosomesNumbers: Object of class `"numeric"` a numeric vector containing the list of chromosomes associated to genomic regions in the GenomicRegions object. Each chromosome is represented just once in increasing order. Please note that chromosomes usually not represented with a number will be comverted to a number as well. e.g. for Human, chromsomomees X and Y will be converted to chromsomes 23 and 24 respectively.

`chromosomesLabels`: Object of class `"character"` a character vector containing the list of chromosomes associated to genomic regions in the GenomicRegions object. Each chromosome is represented just once in the same order as reported in chromosomesNumbers slot. This slot is actually used just to provide a label for each associated chromosome number, in case that some non numeric chromsome is used (e.g. to preserve the correspondence between chr 23 and the actual chr X in Human)

`optionalAnnotations`: Object of class `"matrix"` optional annotations associated to the genomic regions can be managed along with GenomicRegions objects. E.g. the list of GeneSymbol or EntrezGene ids associated to each genomic region can be provided as optional annotation. These additional annotations are not mandatory (the default value for this slot is NULL) The additional annotations must be provided as a matrix of character, with a number of rows equal to the length of "chr", "start" and "end" slots and a number of columns equal to le thength of "optionalAnnotationsHeaders" slot.

`optionalAnnotationsHeaders`: Object of class `"character"` the list of names associated to optional annotations. Please avoid using spaces in annotations names.

`ids`: Object of class `"character"` a character vector of unique identifiers associated to each genomic regions. This is just an optional element of GenomicRegions objects: the default value is NULL.

Methods

GenomicRegions2dataframe `signature(.Object = "GenomicRegions")`: extract genomic regions information as a dataframe object

GenomicRegionsAnnotate `signature(.Object1 = "GenomicRegions", .Object2 = "GenomicAnnotations")`: extract annotations from a GenomicAnnotations object for a set of regions specified as a GenomicRegions object

GenomicRegionsChrNumber `signature(.Object = "GenomicRegions")`: determine the number of chromosomes with genomic regions

GenomicRegionsComparison `signature(.Object1 = "GenomicRegions", .Object2 = "GenomicRegions")`: compare GenomicRegions objects to identify overlaps

GenomicRegionsCreateRegionsIds `signature(.Object = "GenomicRegions")`: generate unique ids for GenomicRegions objects

GenomicRegionsFilter_neg `signature(.Object = "GenomicRegions")`: filter genomic regions to remove selected chromosomes

GenomicRegionsFilter_pos `signature(.Object = "GenomicRegions")`: filter genomic regions to keep selected chromosomes

GenomicRegionsNumber `signature(.Object = "GenomicRegions")`: determine the number of genomic regions

GenomicRegionsSpan `signature(.Object = "GenomicRegions")`: determine the span of each genomic region

GenomicRegionsTotalSpan `signature(.Object = "GenomicRegions")`: determine the total span of genomic regions

initialize `signature(.Object = "GenomicRegions")`: initialize method for GenomicRegions objects

Note

This class is better described in the package vignette

Author(s)

Francesco Ferrari

See Also

GenomicAnnotationsSortAndCleanNA,PREDADataAndResults2dataframe

Examples

```
showClass("GenomicRegions")
```

GenomicRegions2dataframe

extract genomic regions information as a dataframe object

Description

extract genomic regions information as a dataframe object

Usage

```
GenomicRegions2dataframe(GenomicRegionsObject)
```

Arguments

GenomicRegionsObject	Object of class genomic regions

Details

Extract genomic regions information as a dataframe object

Value

A dataframe object

Author(s)

Francesco Ferrari

Examples

```
## Not run:
require(PREDAsampledata)

data(GEanalysisResults)


genomic_regions_UP<-PREDAResults2GenomicRegions(GEanalysisResults
  , qval.threshold=0.05, smoothStatistic.tail="upper",
  smoothStatistic.threshold=0.5)


dataframe_UPregions<-GenomicRegions2dataframe(
genomic_regions_UP[[1]])

## End(Not run)
```

GenomicRegionsAnnotate

extract annotations from a GenomicAnnotations object for a set of regions specified as a GenomicRegions object

---

Description

extract annotations from a GenomicAnnotations object for a set of regions specified as a GenomicRegions object

Usage

```
# GenomicRegionsAnnotate(.Object1, .Object2,
# AnnotationsHeaders=NULL, sep.character="; ",
# complete.inclusion=FALSE, annotationAsRange=FALSE,
# getJustFeaturesNumber=FALSE)

GenomicRegionsAnnotate(.Object1, .Object2, ...)
```

Arguments

.Object1	An object of class GenomicRegions
.Object2	An object of class GenomicAnnotations
...	See below

AnnotationsHeaders: Names of optional annotations fields from GenomicAnnotations object that are used to annotate the GenomicRegions object. Multiple annotation fields can be used

sep.character: Character sequence used to separate annotation features

complete.inclusion: Logical, if TRUE only annotations features entirely covered by one of the genomic regions are considered. (e.g. a gene completely included in the genomic regions from start to end) If FALSE also partial overlapping annotation features are used

annotationAsRange: Logical, if TRUE only the first and last annotation features associated to each the genomic region are returned

getJustFeaturesNumber: Logical, if TRUE only the numbers of annotation features overlapping the genomic regions are returned. If TRUE, only the first element specified with AnnotationsHeaders parameter is considered.

Details

The annotation features overlapping the input genomic regions are used to add optional annotations field to the GenomicRegions object.

If previous optional annotations fields are present, they are preserved as well in the output object

Value

A GenomicRegions object with optionalAnnotations

`GenomicRegionsChrNumber` *determine the number of chromosomes with genomic regions*

Description

determine the number of chromosomes with genomic regions

Usage

```
GenomicRegionsChrNumber(.Object)
```

Arguments

`.Object`	An object of class GenomicRegions

GenomicRegionsComparison *compare GenomicRegions objects to identify overlaps and differences*

Description

compare GenomicRegions objects to identify overlaps and differences

Usage

```
GenomicRegionsComparison(.Object1, .Object2)
```

Arguments

.Object1	An object of Class GenomicRegions
.Object2	An object of Class GenomicRegions

Details

Compare GenomicRegions objects to identify overlaps and differences

Value

A list containing:

`overlapping.regions`
: GenomicRegions object describing the overlapping regions between input object1 and object2

`difference.1.2`
: GenomicRegions object describing the regions from input object1 not overlapping regions from object2

`difference.2.1`
: GenomicRegions object describing the regions from input object2 not overlapping regions from object1

`GenomicRegions1.number`
: Number of genomic regions in input object1

`GenomicRegions2.number`
: Number of genomic regions in input object2

`overlapping.number`
: Number of overlapping genomic regions between input object1 and object2

`GenomicRegions1.totalspan`
: Total span of genomic regions in input object1

`GenomicRegions2.totalspan`
: Total span of genomic regions in input object2

`overlapping.totalspan`
: Total span of overlapping genomic regions between input object1 and object2

`overlap.VS.GenomicRegions1.ratio`
: Ratio between overlapping regions and regions from input object1

`overlap.VS.GenomicRegions2.ratio`
: Ratio between overlapping regions and regions from input object2

Author(s)

Francesco Ferrari

See Also

GenomicRegionsFindOverlap, GenomicRegions

GenomicRegionsCreateRegionsIds

generate unique ids for GenomicRegions objects

Description

generate unique ids for GenomicRegions objects

Usage

```
GenomicRegionsCreateRegionsIds(.Object, ...)
```

Arguments

.Object	An object of class GenomicRegions
...	

GenomicRegionsFilter_neg

filter genomic regions to remove selected chromosomes

Description

filter genomic regions to remove selected chromosomes

Usage

```
# GenomicRegionsFilter_neg(.Object, chrToRemove, chrAsLabels=FALSE, quiet=FALSE)

GenomicRegionsFilter_neg(.Object, ...)
```

Arguments

.Object	An object of class GenomicRegions
...	See below

chrToRemove: List of chromosomes to be removed from the genomic regions object.

chrAsLabels: Logical, TRUE if chromosomes are listed as their character labels, instead of using the numeric indexes

quiet: Logical, if FALSE a message is printed to warn of empty (NULL) result of the filtering selection.

GenomicRegionsFilter_pos

filter genomic regions to keep selected chromosomes

Description

filter genomic regions to keep selected chromosomes

Usage

```
# GenomicRegionsFilter_pos(.Object, chrToRetain, chrAsLabels=FALSE, quiet=FALSE)

GenomicRegionsFilter_pos(.Object, ...)
```

Arguments

.Object	An object of class GenomicRegions
...	See below

chrToRetain: List of chromosomes to be maintained after removing the genomic regions for all the other chromosomes.

chrAsLabels: Logical, TRUE if chromosomes are listed as their character labels, instead of using the numeric indexes

quiet: Logical, if FALSE a message is printed to warn of empty (NULL) result of the filtering selection.

GenomicRegionsFindOverlap *Function to find overlap between GenomicRegions objects*

Description

Function to find overlap between GenomicRegions objects

Usage

```
GenomicRegionsFindOverlap(GenomicRegions1, GenomicRegions2 = NULL)
```

Arguments

GenomicRegions1	Either a GenomicRegions object or a list of GenomicRegions objects
GenomicRegions2	Optiona with default value NULL. Either a GenomicRegions object or a list of GenomicRegions objects.

Details

Input genomic regions object are compared to select overlapping genomic regions that are returned as GenomicRegions objects.

If two single GenomicRegions object are provided, just one comparison is performed and one single GenomicRegions object is returned.

If one single list of GenomicRegions objects is provided as input, then the included GenomicRegions objects are compared to select overlapping GenomicRegions across all of the elements.

If two lists of GenomicRegions objects are provided as input, they must have the same number of elements, because element by element comparison will be performed to identify overlapping GenomicRegions across all of the elements.

Value

Either a single GenomicRegions objec or a list of GenomicRegions objecs.

Author(s)

Francesco Ferrari

See Also

GenomicRegionsComparison, GenomicRegions

Examples

```
## Not run:
require(PREDAsampledata)
data(SODEGIRCNanalysisResults)
data(SODEGIRGEanalysisResults)

SODEGIR_GE_UP<-PREDAResults2GenomicRegions(
SODEGIRGEanalysisResults, qval.threshold=0.05,
smoothStatistic.tail="upper", smoothStatistic.threshold=0.5)

SODEGIR_CN_GAIN<-PREDAResults2GenomicRegions(
SODEGIRCNanalysisResults, qval.threshold=0.01,
smoothStatistic.tail="upper", smoothStatistic.threshold=0.1)

SODEGIR_AMPLIFIED<-GenomicRegionsFindOverlap(SODEGIR_GE_UP,
SODEGIR_CN_GAIN)


## End(Not run)
```

GenomicRegionsFromdataframe

Function to create a GenomiRegions object from a dataframe

Description

Function to create a GenomiRegions object from a dataframe

Usage

```
GenomicRegionsFromdataframe(GenomicRegions_dataframe, ids_column=NULL, chr_column,
start_column, end_column, chromosomesNumbers=NULL,
chromosomesLabels=NULL, chromosomesLabelsInput=NULL)
```

Arguments

GenomicRegions_dataframe	Dataframe object containing the annotations for genomic regions
ids_column	Specify the column from the input dataframe with (optional) ids for genomic regions. Can be specified using column index (numeric) or column name (character).
chr_column	Specify the column from the input dataframe with chromosome annotations fields. Can be specified using column index (numeric) or column name (character).
start_column	Specify the column from the input dataframe with genomic start position for each genomic region. Can be specified using column index (numeric) or column name (character).

`end_column` Specify the column from the input dataframe with genomic end position for each genomic region. Can be specified using column index (numeric) or column name (character).

`chromosomesNumbers` Numeric vector to specify the list of numeric values to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)

`chromosomesLabels` Character vector to specify the list of character labels to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)

`chromosomesLabelsInput` Character vector to specify the list of character labels associated to each chromosome in the input. Particularly useful when non numeric character strings are associated to each chromosome in the input file: e.g. "chr3" for chromosome "3".

Value

An object of class "GenomicRegions"

Author(s)

Francesco Ferrari

See Also

"GenomicRegions"

`GenomicRegionsFromfile` *Function to create a GenomiRegions object from a text file*

Description

Function to create a GenomiRegions object from a text file

Usage

```
GenomicRegionsFromfile(file, ids_column=NULL, chr_column,
start_column, end_column, chromosomesNumbers=NULL,
chromosomesLabels=NULL, chromosomesLabelsInput=NULL, ...)
```

Arguments

`file`	Path to the input txt file containing genomic regions annotations
`ids_column`	Specify the column from the input txt file with (optional) ids for genomic regions. Can be specified using column index (numeric) or column name (character).
`chr_column`	Specify the column from the input txt file with chromosome annotations fields. Can be specified using column index (numeric) or column name (character).
`start_column`	Specify the column from the input txt file with genomic start position for each genomic region. Can be specified using column index (numeric) or column name (character).
`end_column`	Specify the column from the input txt file with genomic end position for each genomic region. Can be specified using column index (numeric) or column name (character).
`chromosomesNumbers`	Numeric vector to specify the list of numeric values to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)
`chromosomesLabels`	Character vector to specify the list of character labels to be associated to each chromosome (especially useful for chromosomes not associated to a number such as chr X or Y)
`chromosomesLabelsInput`	Character vector to specify the list of character labels associated to each chromosome in the input file. Particularly useful when non numeric character strings are associated to each chromosome in the input file: e.g. "chr3" for chromosome "3".
`...`	any other parameter for read.table function that could be useful for parsing the input file, such as "sep", "quote", "header", "na.strings" and other parameters.

Value

An object of class "GenomicRegions"

Author(s)

Francesco Ferrari

See Also

"GenomicRegions"

GenomicRegionsNumber *determine the number of genomic regions*

Description

determine the number of genomic regions

Usage

```
GenomicRegionsNumber(.Object)
```

Arguments

.Object	An object of class GenomicRegions

GenomicRegionsSpan *determine the span of each genomic region*

Description

determine the span of each genomic region

Usage

```
GenomicRegionsSpan(.Object, ...)
```

Arguments

.Object	An object of class GenomicRegions
...	

GenomicRegionsTotalSpan *determine the total span of genomic regions*

Description

determine the total span of genomic regions

Usage

```
GenomicRegionsTotalSpan(.Object, ...)
```

Arguments

.Object	Object of Class GenomicRegions
...	

`getStatisticByName` *extract data for individual analyses using the analysis name*

Description

extract data for individual analyses using the analysis name

Usage

```
# getStatisticByName(.Object, analysisName)

getStatisticByName(.Object, ...)
```

Arguments

`.Object`	An object of class StatisticsForPREDA
`...`	See below
	analysisName: Character name of the analysis to be returned

`MergeStatisticAnnotations2DataForPREDA` *Merge a StatisticsForPREDA and a GenomicAnnotationsForPREDA object into a DataForPREDA object.*

Description

This function merges a StatisticsForPREDA and a GenomicAnnotationsForPREDA object into a DataForPREDA object

Usage

```
MergeStatisticAnnotations2DataForPREDA(StatisticsForPREDAObject,
GenomicAnnotationsForPREDAObject, sortAndCleanNA = FALSE, quiet =
FALSE, MedianCenter = FALSE)
```

Arguments

`StatisticsForPREDAObject`	An object of class StatisticsForPREDA
`GenomicAnnotationsForPREDAObject`	An object of class GenomicAnnotationsForPREDA
`sortAndCleanNA`	Logical, if TRUE, genomic annotations are sorted for chromosome and genomic position then ids with NA positinal annotations are removed
`quiet`	Logical, if TRUE messages reporting the number of unmatched ids are suppressed.
`MedianCenter`	Logical, if TRUE data are normalized per median sample.

Value

An object of class DataForPREDA

Author(s)

Francesco Ferrari

`PREDADataAndResults-class`

Class "PREDADataAndResults" is used to manage the PREDA analysis output

---

Description

This class is used to manage the PREDA analysis output along with corresponding input data

Objects from the Class

Objects can be created by calls of the form `new("PREDADataAndResults", ids, chr, start, end, strand, chromosomesNumbers, chromosomesLabels, position, optionalAnnotations, optionalAnnotationsHeaders, analysesNames, testedTail, smoothStatistic, pvalue, qvalue, statistic)`.

Slots

`analysesNames`**:** Object of class `"character"` a character vector of unique names associated to each column of smoothStatistic, pvalue and qvalue matrices. This is just a name that is used to identify each analysis.

`testedTail`**:** Object of class `"character"` a character describing what tail of the statistic distribution will be analyzed during PREDA analysis. Possible values are "upper", "lower" or "both". Anyway we strongly recommend using PREDA analysis only

`smoothStatistic`**:** Object of class `"matrix"` a numeric matrix containing smoothed observed statistics as obtained from PREDA analysis. The smoothed statistics must be provided as a matrix of numeric values, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "analysesNames" slot.

`pvalue`**:** Object of class `"matrix"` a numeric matrix containing unadjusted gene-centered pvalues as obtained from PREDA analysis. The pvalue matrix must be provided as a matrix of numeric values, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "analysesNames" slot.

`qvalue`**:** Object of class `"matrix"` a numeric matrix containing adjusted gene-centered pvalues as obtained from PREDA analysis: i.e. usually FDR adjusted pvalues, but other multiple testing methods could be adopted as well The qvalue matrix must be provided as a matrix of numeric values, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "analysesNames" slot.

`position`: Object of class `"integer"` a numeric vector of reference genomic positions that will be associated and used for each genomic feature under investigation for smoothing data during PREDA analysis.

`ids`: Object of class `"character"` a character vector of unique identifiers for the genomic features under investigation

`chr`: Object of class `"integer"` a numeric vector representing the chromosome where each ids is mapped. Please note that chromosome usually not represented with a number must will be comverted to a number as well. e.g. for Human, chromsomomees X and Y will be converted to chromsomes 23 and 24 respectively. User defined options will allow this conversion during GenomicAnnotations objects initialization.

`start`: Object of class `"integer"` a numeric vector of start genomic position for each genomic feature under investigation (i.e. gene, transcript, SNP or other elements).

`end`: Object of class `"integer"` a numeric vector of end genomic position for each genomic feature under investigation (i.e. gene, transcript, SNP or other elements).

`strand`: Object of class `"numeric"` a numeric vector of strand genomic position for each genomic feature under investigation: value 1 is used for "plus" (forward) strand and value -1 for "minus" (reverse) strand. User defined options will allow the conversion to this format during GenomicAnnotations objects initialization.

`chromosomesNumbers`: Object of class `"numeric"` a numeric vector containing the list of chromosomes for which genomic annotations are provided in the GenomicAnnotations object. Each chromosome is represented just once in increasing order. Please note that chromosome usually not represented with a number must will be comverted to a number as well. e.g. for Human, chromsomomees X and Y will be converted to chromsomes 23 and 24 respectively.

`chromosomesLabels`: Object of class `"character"` a character vector containing the list of chromosomes for which genomic annotations are provided in the GenomicAnnotations object. Each chromosome is represented just once in the same order as reported in chromosomesNumbers slot. This slot is actually used just to provide a label for each associated chromosome number, in case that some non numeric chromsome is used (e.g. to preserve the correspondence between chr 23 and the actual chr X in Human)

`optionalAnnotations`: Object of class `"matrix"` optional annotations associated to the genomic features can be managed along with genomic positions annotations. E.g. GeneSymbol or EntrezGene ids can be associated to gene realted GenomicAnnotaitons objects. These additional annotations are not mandatory (the default value for this slot is NULL) The additional annotations must be provided as a matrix of character, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "optionalAnnotationsHeaders" slot.

`optionalAnnotationsHeaders`: Object of class `"character"` character vector containing the names associated to optional annotations. Please avoid using spaces in annotations names.

`statistic`: Object of class `"matrix"` a numeric matrix containing gene-centered statistics (or statistics on genomic data centered on other genomic features under investigation). The statistics must be provided as a matrix of numeric values, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "analysesNames" slot.

Extends

Class `"PREDAResults"`, directly. Class `"DataForPREDA"`, directly. Class `"GenomicAnnotationsForPREDA"`, by class "PREDAResults", distance 2. Class `"StatisticsForPREDA"`, by class "DataForPREDA", distance 2. Class `"GenomicAnnotations"`, by class "PREDAResults", distance 3.

Methods

GenomicAnnotationsSortAndCleanNA `signature(.Object = "PREDADataAndResults")`: sort annotations according to selected chromosomes and to remove genes containing any NA annotation field

initialize `signature(.Object = "PREDADataAndResults")`: initialize method for PREDADataAndResults objects

PREDADataAndResults2dataframe `signature(.Object = "PREDADataAndResults")`: extract data and annotations as a dataframe with probeids as rownames

Note

This class is better described in the package vignette

Author(s)

Francesco Ferrari

See Also

"GenomicAnnotations", "GenomicAnnotationsForPREDA", "StatisticsForPREDA", "DataForPREDA", "PREDAResults", GenomicAnnotationsSortAndCleanNA,PREDADataAndResults2dataframe

Examples

```
showClass("PREDADataAndResults")
```

PREDADataAndResults2dataframe

extract data and annotations as a dataframe with probeids as rownames

---

Description

extract data and annotations as a dataframe with probeids as rownames

Usage

```
PREDADataAndResults2dataframe(.Object)
```

Arguments

.Object	An object of class PREDADataAndResults

PREDAResults-class *Class "PREDAResults" ~is used to manage the PREDA analysis output*

Description

this class is used to manage the basic PREDA analysis output including smoothened statistic, pvalues and qvalues.

Objects from the Class

Objects can be created by calls of the form `new("PREDAResults", ids, chr, start, end, strand, chromosomesNumbers, chromosomesLabels, position, optionalAnnotations, optionalAnnotationsHeaders, analysesNames, testedTail, smoothStatistic, pvalue, qvalue)`.

Slots

`analysesNames`: Object of class `"character"` a character vector of unique names associated to each column of smoothStatistic, pvalue and qvalue matrices. This is just a name that is used to identify each analysis.

`testedTail`: Object of class `"character"` a character describing what tail of the statistic distribution will be analyzed during PREDA analysis. Possible values are "upper", "lower" or "both". Anyway we strongly recommend using PREDA analysis only

`smoothStatistic`: Object of class `"matrix"` a numeric matrix containing smoothed observed statistics as obtained from PREDA analysis. The smoothed statistics must be provided as a matrix of numeric values, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "analysesNames" slot.

`pvalue`: Object of class `"matrix"` a numeric matrix containing unadjusted gene-centered pvalues as obtained from PREDA analysis. The pvalue matrix must be provided as a matrix of numeric values, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "analysesNames" slot.

`qvalue`: Object of class `"matrix"` a numeric matrix containing adjusted gene-centered pvalues as obtained from PREDA analysis: i.e. usually FDR adjusted pvalues, but other multiple testing methods could be adopted as well The qvalue matrix must be provided as a matrix of numeric values, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "analysesNames" slot.

`position`: Object of class `"integer"` a numeric vector of reference genomic positions that will be associated and used for each genomic feature under investigation for smoothing data during PREDA analysis.

`ids`: Object of class `"character"` a character vector of unique identifiers for the genomic features under investigation

`chr`: Object of class `"integer"` a numeric vector representing the chromosome where each ids is mapped. Please note that chromosome usually not represented with a number must will be comverted to a number as well. e.g. for Human, chromsomomees X and Y will be converted to chromsomes 23 and 24 respectively. User defined options will allow this conversion during GenomicAnnotations objects initialization.

`start`: Object of class `"integer"` a numeric vector of start genomic position for each genomic feature under investigation (i.e. gene, transcript, SNP or other elements).

`end`: Object of class `"integer"` a numeric vector of end genomic position for each genomic feature under investigation (i.e. gene, transcript, SNP or other elements).

`strand`: Object of class `"numeric"` a numeric vector of strand genomic position for each genomic feature under investigation: value 1 is used for "plus" (forward) strand and value -1 for "minus" (reverse) strand. User defined options will allow the conversion to this format during GenomicAnnotations objects initialization.

`chromosomesNumbers`: Object of class `"numeric"` a numeric vector containing the list of chromosomes for which genomic annotations are provided in the GenomicAnnotations object. Each chromosome is represented just once in increasing order. Please note that chromosome usually not represented with a number must will be comverted to a number as well. e.g. for Human, chromsomomees X and Y will be converted to chromsomes 23 and 24 respectively.

`chromosomesLabels`: Object of class `"character"` a character vector containing the list of chromosomes for which genomic annotations are provided in the GenomicAnnotations object. Each chromosome is represented just once in the same order as reported in chromosomesNumbers slot. This slot is actually used just to provide a label for each associated chromosome number, in case that some non numeric chromsome is used (e.g. to preserve the correspondence between chr 23 and the actual chr X in Human)

`optionalAnnotations`: Object of class `"matrix"` optional annotations associated to the genomic features can be managed along with genomic positions annotations. E.g. GeneSymbol or EntrezGene ids can be associated to gene realted GenomicAnnotaitons objects. These additional annotations are not mandatory (the default value for this slot is NULL) The additional annotations must be provided as a matrix of character, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "optionalAnnotationsHeaders" slot.

`optionalAnnotationsHeaders`: Object of class `"character"` character vector containing the names associated to optional annotations. Please avoid using spaces in annotations names.

Extends

Class `"GenomicAnnotationsForPREDA"`, directly. Class `"GenomicAnnotations"`, by class "GenomicAnnotationsForPREDA", distance 2.

Methods

GenomicAnnotationsSortAndCleanNA `signature(.Object = "PREDAResults")`: sort annotations according to selected chromosomes and to remove genes containing any NA annotation field

initialize `signature(.Object = "PREDAResults")`: initialize method for PREDAResults objects

PREDAResults2dataframe `signature(.Object = "PREDAResults")`: extact preda results statistics as a data frame object

PREDAResults2GenomicRegions `signature(.Object = "PREDAResults")`: identify significant genomic regions from a PREDAResults object

PREDAResults2GenomicRegionsSingle `signature(.Object = "PREDAResults")`: identify significant genomic regions from a single analysis in a PREDAResults object

PREDAResults2PREDADataAndResults `signature(.Object = "PREDAResults")`: merge PREDAResults and input statistics to create a PREDADataAndResults object

PREDAResultsGetObservedFlags `signature(.Object = "PREDAResults")`: extract genomic positions with significant alterations as a matrix of flags from a PREDAResults object

Note

This class is better described in the package vignette

Author(s)

Francesco Ferrari

See Also

"GenomicAnnotations", "GenomicAnnotationsForPREDA", GenomicAnnotationsSortAndCleanNA, PREDAResults2dataframe, PREDAResults2GenomicRegions, PREDAResults2GenomicRegionsSingle, PREDAResults2PREDADataAndResults, PREDAResultsGetObservedFlags

Examples

```
showClass("PREDAResults")
```

PREDAResults2dataframe

extact preda results statistics as a data frame object

Description

extact preda results statistics as a data frame object

Usage

```
PREDAResults2dataframe(.Object)
```

Arguments

`.Object`	An object of class PREDAResults

PREDAResults2GenomicRegions
identify significant genomic regions from a PREDAResults object

Description

identify significant genomic regions from a PREDAResults object

Usage

```
# PREDAResults2GenomicRegions(.Object, qval.threshold=0.05,
# use.referencePositions=TRUE, smoothStatistic.tail=NULL,
# smoothStatistic.threshold=NULL)

PREDAResults2GenomicRegions(.Object, ...)
```

Arguments

.Object	Object of class PREDAResults or PREDADataAndResults
...	See below

qval.threshold: q-value threshold used to identify significant genomic regions

use.referencePositions: Logical, if TRUE the input reference positions used for PREDA analysis wil be used to identify significant genomic regions boundaries as well.

smoothStatistic.tail: Possible values are "upper" or "lower". This parameter specify if only one tail of the smoothed statististic distribution must be considered. If it is NULL, both tails are used and smoothStatistic.threshold is ignored.

smoothStatistic.threshold: Threshold on smoothStatistic values to select significant genomic regions.

Details

A list og genomic regions objects is returned: one GenomicRegions object for each analysis in the input PREDAresults.

A NULL element is included in the output list whenever no siginifcant regions are identified.

Value

A list of genomic regions objects

Author(s)

Francesco Ferrari

Examples

```
## Not run:
require(PREDAsampledata)

data(GEanalysisResults)

genomic_regions_UP<-PREDAResults2GenomicRegions(GEanalysisResults
, qval.threshold=0.05, smoothStatistic.tail="upper",
smoothStatistic.threshold=0.5)


## End(Not run)
```

`PREDAResults2GenomicRegionsSingle` *identify significant genomic regions from a single analysis in a PREDAResults object*

Description

identify significant genomic regions from a single analysis in a PREDAResults object

Usage

```
# PREDAResults2GenomicRegionsSingle(.Object,
# qval.threshold=0.05, analysisName=NULL,
# use.referencePositions=TRUE, smoothStatistic.tail=NULL,
# smoothStatistic.threshold=NULL)

PREDAResults2GenomicRegionsSingle(.Object, ...)
```

Arguments

`.Object`	Object of class PREDAResults or PREDADataAndResults
`...`	See below
	qval.threshold: q-value threshold used to identify significant genomic regions
	analysisName: name of the analysis to be considered
	use.referencePositions: Logical, if TRUE the input reference positions used for PREDA analysis wil be used to identify significant genomic regions boundaries as well.
	smoothStatistic.tail: Possible values are "upper" or "lower". This parameter specify if only one tail of the smoothed statististic distribution must be considered. If it is NULL, both tails are used and smoothStatistic.threshold is ignored.
	smoothStatistic.threshold: Threshold on smoothStatistic values to select significant genomic regions.

PREDAResults2PREDADataAndResults
merge PREDAResults and input statistics to create a PREDADataAndResults object

Description

merge PREDAResults and input statistics to create a PREDADataAndResults object

Usage

```
# PREDAResults2PREDADataAndResults(.Object, statistic)

PREDAResults2PREDADataAndResults(.Object, ...)
```

Arguments

.Object	An object of class PREDAResults
...	See below
	statistic: A matrix containing input statistics

PREDAResultsGetObservedFlags
extract genomic positions with significant alterations as a matrix of flags from a PREDAResults object

Description

extract genomic positions with significant alterations as a matrix of flags from a PREDAResults object

Usage

```
# PREDAResultsGetObservedFlags(.Object, qval.threshold=0.05,
# smoothStatistic.tail=NULL, smoothStatistic.threshold=NULL,
# null.value=0, significant.value=1)

PREDAResultsGetObservedFlags(.Object, ...)
```

Arguments

.Object	An object of class PREDAResults or PREDADataAndResults
...	See below
	qval.threshold: q-value threshold used to identify significant genomic positions
	smoothStatistic.tail: Possible values are "upper" or "lower". This parameter specify if only one tail of the smoothed statististic distribution must be considered. If it is NULL, both tails are used and smoothStatistic.threshold is ignored.
	smoothStatistic.threshold: Threshold on smoothStatistic values to select significant genomic regions.
	null.value: Value (flag) assigned to not significant positions
	significant.value: Value (flag) assigned to significant positions

---

PREDA_main *function performing the core of PREDA analysis*

Description

function performing the core of PREDA analysis

Usage

```
PREDA_main(inputDataForPREDA, outputGenomicAnnotationsForPREDA
=NULL, nperms = 10000, verbose = TRUE, parallelComputations =
FALSE, multTestCorrection = "fdr", permutePerChromosome = FALSE,
blocksize = 10, permuteStatisticSign = FALSE, smoothMethod =
"lokern_scaledBandwidth_repeated", force = FALSE,
lokern_scaledBandwidthFactor = 2, limit.analysis = NULL)
```

Arguments

inputDataForPREDA	A Data for PREDA object
outputGenomicAnnotationsForPREDA	A GenomicAnnotationsForPREDA object. If NULL, GenomicsAnnotations for output data are obtained from inputDataForPREDA
nperms	Number of permutations performed in PREDA analysis.
verbose	Logical, if TRUE some messages are printed concenrning the advancement of the analysis.
parallelComputations	Logical, if TRUE Rmpi is used to spawn slave processes, thus using parallel computing to speedup the analysis.

`multTestCorrection`
: Multiple testing correction that will be adopted to correct the statistic p-values. Possible values are "fdr", for benjamini and Hochberg multiple testing correction and "qvalue" for p-values correction performed with qvalue package.

`permutePerChromosome`
: Logical, if TRUE data parmutations are perfored separatedly for each chromsoome. In most cases the default value (FALSE) is preferable to avoid biases related to specific chromosomes extreme alterations.

`blocksize`
: A parameter used to tune parallel computations if parallelComputations is TRUE. This is actually the number of permutations performed on each slave process before every communication with master process.

  This is useftul to reduce the numebr of network communications when slow communicatinos are established among slave processes.

`permuteStatisticSign`
: Logical, if TRUE statistics signs are permuted instead of permuting data along chromsomal position.

`smoothMethod`
: The deafault smoothing metod used in the PREDA_main function is lokern smoothing with scaled bandwidth, using a scaling factor equal to 2.

  Possible values are "lokern", for standard lokern smoothing, "quantsmooth", "spline" and "runningmean.x", where x is a user defined value for the number of adjacent data points using for running mean smoothing.

`force`
: Logical, if TRUE force skipping quantsmooth control on number of data points. Singe quantsmooth is very slow with a high number of inpuit data, a check stopping computation with more than 2000 data points in one or more chromosome was introduced. This aprameter allow skippin this security check.

`lokern_scaledBandwidthFactor`
: Factor of scaling for lokern estimated bandwidths

`limit.analysis`
: Vector (numeric or character representing analyses names) to limit the output of preda analysis to a subset of input analyses.

Details

See supplementary material about PREDA method

Value

If outputGenomicAnnotationsForPREDA is NULL, a PREDADataAndResults object is returned. Otherwise a PREDAResults object is returned instead

Author(s)

Francesco Ferrari

See Also

Supplementary information about PREDA method

Examples

```
#See examples in PREDA tutorial
```

`preprocessingGE` *Wrapper function for gene expression data preprocessing for differential expression analysis with PREDA*

Description

Wrapper function for gene expression data preprocessing for differential expression analysis with PREDA

Usage

```
preprocessingGE(SampleInfoFile = NULL, CELfiles_dir = NULL,
AffyBatchInput = NULL, custom_cdfname, arrayNameColumn = NULL,
sampleNameColumn = NULL, classColumn,
referenceGroupLabel, statisticType, optionalAnnotations = NULL,
retain.chrs = NULL, reference_position_type = "median",
testedTail = "both")
```

Arguments

`SampleInfoFile`	Path to sample info file
`CELfiles_dir`	Path to directory containing raw CEL data files for Affymetrix arrays
`AffyBatchInput`	Alternatively input raw data can be provided as an AffyBatch object. In this case sample classes will be inferred from phenodata contained in AffyBatch object. In particular classColumn parameter will refer to the column in pData(AffyBatchInput) object.
`custom_cdfname`	Specify the cdf library to be used for data preprocessing
`arrayNameColumn`	Column of sampleinfo file containing the name of raw data (CEL) files
`sampleNameColumn`	Column of sampleinfo file containing the name to be used for samples labels
`classColumn`	Column of sampleinfo file containing the label of sample classes. If input raw data are provided as an AffyBatch object, this parameter refers intead to the column in pData(AffyBatchInput) object.
`referenceGroupLabel`	Specify which class label is used for the reference sample used in computing statistics for differential expression.
`statisticType`	Stastistic for differential expression that is computed on input data. Possible values are "tstatistic", "FC" (Fold Change), "FCmedian" (fold change computed on medians)

`optionalAnnotations`
: Character vector to select additional annotations fields to be included into the GenomicAnnotations object.

`retain.chrs`
: Numeric vector, containing the list of chromosomes selected for the output GenomicAnnotations object. E.g. set retain.chrs=1:22 to limit the GenomicAnnotations object to chromosomes from 1 to 22. This might be ueseful to limit GenomiAnnotations objects to autosomic chromosomes.

`reference_position_type`
: Specify which genomic coordinate must be used as reference position for PREDA analysis. Possible values are "start", "end", "median", "strand.start" or "strand.end".

`testedTail`
: Specify what tail of the distribution will be tested for significantly extreme values in PREDA analysis. Possible values are "both", "upper" or "lower".

Details

Preprocess raw (CEL) files for Affymetrix gene expression arrays using user defined CDF libraries and RMA normalization. Then statistics for differential expression are computed. Then annotations are retrieved from the corresponding annotation library.

Please note this function is a user-friendly preprocessing function for Affy gene expression microarrays. Step by step preprocessing functions can be used with any other platform.

Value

A DataForPREDA object is returned.

Author(s)

Francesco Ferrari

See Also

DataForPREDA

Examples

```
## Not run:

require("PREDAsampledata")
CELfilesPath <- system.file("sampledata", "GeneExpression",
package = "PREDAsampledata")
infofile <- file.path(CELfilesPath , "sampleinfoGE_PREDA.txt")
sampleinfo<-read.table(infofile, sep="\t", header=TRUE)


GEDataForPREDA<-preprocessingGE(SampleInfoFile=infofile,
CELfiles_dir=CELfilesPath,
custom_cdfname="hgu133plus2",
arrayNameColumn=1,
sampleNameColumn=2,
classColumn="Class",
```

```
referenceGroupLabel="normal",
statisticType="tstatistic",
optionalAnnotations=c("SYMBOL", "ENTREZID"),
retain.chrs=1:22
)

## End(Not run)
```

SODEGIRpreprocessingGE *Wrapper function for gene expression data preprocessing for SODEGIR analysis*

Description

Wrapper function for gene expression data preprocessing for SODEGIR analysis

Usage

```
SODEGIRpreprocessingGE(SampleInfoFile = NULL, CELfiles_dir = NULL,
AffyBatchInput = NULL, custom_cdfname, arrayNameColumn = NULL,
sampleNameColumn = NULL, classColumn,
referenceGroupLabel, statisticType, optionalAnnotations = NULL,
retain.chrs = NULL, reference_position_type = "median",
testedTail = "both", singleSampleOutput = TRUE,
varianceAll=FALSE)
```

Arguments

SampleInfoFile	Path to sample info file
CELfiles_dir	Path to directory containing raw CEL data files for Affymetrix arrays
AffyBatchInput	Alternatively input raw data can be provided as an AffyBatch object. In this case sample classes will be inferred from phenodata contained in AffyBatch object. In particular classColumn parameter will refer to the column in pData(AffyBatchInput) object.
custom_cdfname	Specify the cdf library to be used for data preprocessing
arrayNameColumn	Column of sampleinfo file containing the name of raw data (CEL) files
sampleNameColumn	Column of sampleinfo file containing the name to be used for samples labels
classColumn	Column of sampleinfo file containing the label of sample classes. If input raw data are provided as an AffyBatch object, this parameter refers intead to the column in pData(AffyBatchInput) object.
referenceGroupLabel	Specify which class label is used for the reference sample used in computing statistics for differential expression.

`statisticType` Stastistic for differential expression that is computed on input data. Possible values are "tstatistic", "FC" (Fold Change), "FCmedian" (fold change computed on medians)

`optionalAnnotations`
Character vector to select additional annotations fields to be included into the GenomicAnnotations object.

`retain.chrs` Numeric vector, containing the list of chromosomes selected for the output GenomicAnnotations object. E.g. set retain.chrs=1:22 to limit the GenomicAnnotations object to chromosomes from 1 to 22. This might be ueseful to limit GenomiAnnotations objects to autosomic chromosomes.

`reference_position_type`
Specify which genomic coordinate must be used as reference position for PREDA analysis. Possible values are "start", "end", "median", "strand.start" or "strand.end".

"strand.start" is strand specific start: i.e. start on positive strand but end on negative strand. "strand.end" is strand specific end.

`testedTail` Specify what tail of the distribution will be tested for significantly extreme values in PREDA analysis. Possible values are "both", "upper" or "lower".

`singleSampleOutput`
Logical, if TRUE a statistic comparing each sample with the reference group is computed.

`varianceAll` This parameter affect the computation only when singleSampleOutput is TRUE.

varianceAll is itself a logical parameter. If TRUE, all pathological (e.g. tumor) samples and all normal (reference) samples are used to estimate variance in the comparison of individual pathological samples to the normal reference, as described in the original SODEGIR apper by Bicciato et al. (Nucleic Acids Res. 2009).

The original SODEGIR statistic for Gene Expression was based on the SAM score. However, since July 2018 the original samr package is no more available in CRAN. Therefore in the current PREDA version the varianceAll=TRUE and singleSampleOutput=TREU can't be used with SAM. When singleSampleOutput is TRUE and a different statisticType is used, the variance is actually computed using only the normal (reference) samples.

If FALSE (default value), the computation of statistics for single sample VS reference comparisons only take into account the variance in the reference group of samples.

Details

Preprocess raw (CEL) files for Affymetrix gene expression arrays using user defined CDF libraries and RMA normalization.

Then statistics for differential expression are computed comparing each sample with the reference group.

Then annotations are retrieved from the corresponding annotation library.

Please note this function is a user-friendly preprocessing function for Affy gene expression microarrays. Step by step preprocessing functions can be used with any other platform.

Value

A DataForPREDA object is returned.

Author(s)

Francesco Ferrari

References

Silvio Bicciato, Roberta Spinelli, Mattia Zampieri, Eleonora Mangano, Francesco Ferrari, Luca Beltrame, Ingrid Cifola, Clelia Peano, Aldo Solari, and Cristina Battaglia. A computational procedure to identify significant overlap of differentially expressed and genomic imbalanced regions in cancer datasets. Nucleic Acids Res, 37(15):5057-70, August 2009.

See Also

preprocessingGE, DataForPREDA

Examples

```
  ## Not run:
require(PREDAsampledata)

CELfilesPath <- system.file("sampledata", "GeneExpression",
package = "PREDAsampledata")

infofile <- file.path(CELfilesPath , "sampleinfoGE_PREDA.txt")

SODEGIRGEDataForPREDA<-SODEGIRpreprocessingGE(SampleInfoFile=
infofile,
CELfiles_dir=CELfilesPath,
custom_cdfname="hgu133plus2",
arrayNameColumn=1,
sampleNameColumn=2,
classColumn="Class",
referenceGroupLabel="normal",
statisticType="tstatistic",
optionalAnnotations=c("SYMBOL", "ENTREZID"),
retain.chrs=1:22
)



## End(Not run)
```

SODEGIR_GEstatistics *Wrapper function for gene expression statistics preprocessing for SODEGIR analysis*

Description

Wrapper function for gene expression statistics preprocessing for SODEGIR analysis.

Usage

```
# SODEGIR_GEstatistics(.Object, pData_classColumn=NULL,
# referenceGroupLabel=NULL,
# statisticType=c("tstatistic", "FC", "FCmedian", "eBayes"),
# singleSampleOutput=TRUE, varianceAll=FALSE)

SODEGIR_GEstatistics(.Object, ...)
```

Arguments

`.Object` An object of class ExpressionSet containing gene expression input data

`...` See below

pData_classColumn: Column of phenoData slot from the ExpressionSet object, containing the label of sample classes

referenceGroupLabel: Specify which class label is used for the reference sample used in computing statistics for differential expression.

statisticType: Stastistic for differential expression that is computed on input data. Possible values are "tstatistic", "FC" (Fold Change), "FCmedian" (fold change computed on medians)

singleSampleOutput: Logical, if TRUE a statistic comparing each sample with the reference group is computed.

varianceAll: This parameter affect the computation only when singleSampleOutput is TRUE.
varianceAll is itself a logical parameter. If TRUE, all pathological (e.g. tumor) samples and all normal (reference) samples are used to estimate variance in the comparison of individual pathological samples to the normal reference, as described in the original SODEGIR apper by Bicciato et al. (Nucleic Acids Res. 2009).
The original SODEGIR statistic for Gene Expression was based on the SAM score. However, since July 2018 the samr package is no more available in CRAN. Therefore in the current PREDA version the varianceAll=TRUE parameter can't be used as SAM is not available. When singleSampleOutput is TRUE and a different statisticType is used, the variance is actually computed using only the normal (reference) samples.
If FALSE (default value), the computation of statistics for single sample VS reference comparisons only take into account the variance in the reference group of samples.

Details

Using an ExpressionSet object as input, statistics for differential expression are computed comparing each sample with the reference group.

Value

The output is returned as a matrix.

Author(s)

Francesco Ferrari

References

Silvio Bicciato, Roberta Spinelli, Mattia Zampieri, Eleonora Mangano, Francesco Ferrari, Luca Beltrame, Ingrid Cifola, Clelia Peano, Aldo Solari, and Cristina Battaglia. A computational procedure to identify significant overlap of differentially expressed and genomic imbalanced regions in cancer datasets. Nucleic Acids Res, 37(15):5057-70, August 2009.

See Also

preprocessingGE, SODEGIRpreprocessingGE, ExpressionSet

StatisticsForPREDA-class

Class "StatisticsForPREDA" is used to manage the datamatrix containing statistics for PREDA analyses

---

Description

This class is used to manage the datamatrix containing statistics for PREDA analyses: i.e. the gene (or other genomic feature) centered statistics accounting for differential expression (or for the other type of variation under investigation)

Objects from the Class

Objects can be created by calls of the form `new("StatisticsForPREDA", ids, statistic, analysesNames, testedTail)`.

Slots

`ids`: Object of class `"character"` a character vector of unique identifiers for the genomic features under investigation

`statistic`: Object of class `"matrix"` a numeric matrix containing gene-centered statistics (or statistics on genomic data centered on other genomic features under investigation). The statistics must be provided as a matrix of numeric values, with a number of rows equal to the length of "ids" slot and a number of columns equal to the length of "analysesNames" slot.

`analysesNames`: Object of class `"character"` a character vector of unique names associated to each column of statistic matrix. This is just a name that will be used to identify each analysis.

`testedTail`: Object of class `"character"` a character describing what tail of the statistic distribution will be analyzed during PREDA analysis. Possible values are "upper", "lower" or "both". Anyway we strongly recommend using PREDA analysis only for statistics on genomic data with a symmetric distribution around zero.

Methods

analysesNames `signature(.Object = "StatisticsForPREDA")`: get the names of the analyses in the StatisticsForPREDA object

getStatisticByName `signature(.Object = "StatisticsForPREDA")`: extract data for individual analyses using the analysis name

initialize `signature(.Object = "StatisticsForPREDA")`: initialize method for StatisticsForPREDA objects

StatisticsForPREDA2dataframe `signature(.Object = "StatisticsForPREDA")`: extract data as a dataframe with probeids as rownames

StatisticsForPREDAFilterColumns_neg `signature(.Object = "StatisticsForPREDA")`: filter statistics to remove selected analyses

StatisticsForPREDAFilterColumns_pos `signature(.Object = "StatisticsForPREDA")`: filter statistics to keep selected analyses

Note

This class is better described in the package vignette

Author(s)

Francesco Ferrari

See Also

`"DataForPREDA"`, `analysesNames`,`getStatisticByName` `StatisticsForPREDA2dataframe`, `StatisticsForPREDAFilter`

Examples

```
showClass("StatisticsForPREDA")
```

`StatisticsForPREDA2dataframe`
extract data as a dataframe with probeids as rownames

Description

extract data as a dataframe with probeids as rownames

Usage

```
StatisticsForPREDA2dataframe(.Object)
```

Arguments

`.Object`	An object of class StatisticsForPREDA

`StatisticsForPREDAFilterColumns_neg`
filter statistics to remove selected analyses

Description

filter statistics to remove selected analyses

Usage

```
# StatisticsForPREDAFilterColumns_neg(.Object, analysesToRemove,
# analysesAsNames=FALSE)

StatisticsForPREDAFilterColumns_neg(.Object, ...)
```

Arguments

`.Object`	An object of class StatisticsForPREDA
`...`	See below

analysesToRemove: Analysis statistics columns to be removed after filtering

analysesAsNames: Logical, if TRUE analyses are listed as their character names. If FALSE they can be listed as numeric indexes.

`StatisticsForPREDAFilterColumns_pos`
filter statistics to keep selected analyses

Description

filter statistics to keep selected analyses

Usage

```
# StatisticsForPREDAFilterColumns_pos(.Object, analysesToRetain,
# analysesAsNames=FALSE)

StatisticsForPREDAFilterColumns_pos(.Object, ...)
```

Arguments

`.Object`	An object of class StatisticsForPREDA
`...`	See below
	analysesToRetain: Analysis statistics columns to be retained after filtering
	analysesAsNames: Logical, if TRUE analyses are listed as their character names. If FALSE they can be listed as numeric indexes.

`StatisticsForPREDAFromdataframe`
Function to create a StatisticsForPREDA objet from a dataframe

---

Description

Function to create a StatisticsForPREDA objet from a dataframe

Usage

```
StatisticsForPREDAFromdataframe(StatisticsForPREDA_dataframe, ids_column = NULL,
statistic_columns = NULL, analysesNames = NULL, testedTail =
c("upper", "lower", "both"))
```

Arguments

`StatisticsForPREDA_dataframe`	Input dataframe containing statistics on genomics data.
`ids_column`	Specify the column from the input dataframe with gene (or other genomic features) ids. Can be specified using column index (numeric) or column name (character).

`statistic_columns`
: Specify the column (or columns) from the input dataframe with gsta.enomic data statistics that will be included in the statisticsForPREDA object. Can be specified using column index (numeric) or column name (character).
If NULL (default), all columns excluding ids_column will be considered as input statistics

`analysesNames`
: Names (labels) to be associated to each input statistic. If NULL the column names for statistics_columns will be used.

`testedTail`
: Specify what tail of the distribution will be tested for significantly extreme values in PREDA analysis. Possible values are "both", "upper" or "lower".

`...`
: any other parameter for read.table function that could be useful for parsing the input file, such as "sep", "quote", "header", "na.strings" and other parameters.

Details

A dataframe is parsed and a statisticsForPREDA object is built using contained data.

Value

A statisticsForPREDA object

Author(s)

Francesco Ferrari

See Also

StatisticsForPREDA

Examples

```
  ## Not run:
require(PREDAsampledata)

CNdataPath <- system.file("sampledata", "CopyNumber", package =
"PREDAsampledata")

CNdataFile <- file.path(CNdataPath , "CNAG_data_PREDA.txt")

CNannotationFile <- file.path(CNdataPath , "SNPAnnot100k.csv")

CNStatisticsForPREDA<-StatisticsForPREDAFromdataframe(file=CNdataFile,
ids_column="AffymetrixSNPsID", testedTail="both", sep="\t",
header=TRUE)


## End(Not run)
```

`statisticsForPREDAfromEset` *function to compute a statisticsForPREDA object from an ExpressionSet object*

Description

function to compute a statisticsForPREDA object from an ExpressionSet object

Usage

```
# statisticsForPREDAfromEset(.Object, pData_classColumn=NULL,
# statisticType=NULL, logged=TRUE, referenceGroupLabel=NULL,
# classVector=NULL, testedTail="both")

statisticsForPREDAfromEset(.Object, ...)
```

Arguments

`.Object`	Object of class ExpressionSet
`...`	See below

pData_classColumn: Column from pData(.Object) containig the labels for different samples classes.

statisticType: Stastistic for differential expression that is computed on input data. Possible values are "tstatistic", "FC" (Fold Change), "FCmedian" (fold change computed on medians)

logged: Logical value (default TRUE) to specify if the input data are logged (Log2). This parameter will influence the computation of statistics.

referenceGroupLabel: Specify which class label is used for the reference sample used in computing statistics for differential expression.

classVector: If pData_classColumn is NULL then a vector specifying the sample classes is required and can be provided with classVector parameter

testedTail: Specify what tail of the distribution will be tested for significantly extreme values in PREDA analysis. Possible values are "both", "upper" or "lower". Default value is "both".

Details

An object of class ExpressionSet is used as input and gene centered statistics for differential expression are computed on the contained data. The computed statistics are used to build a StatisticsForPREDA object

Value

An object of class StatisticsForPREDA

Author(s)

Francesco Ferrari

See Also

"StatisticsForPREDA"

Examples

```
  ## Not run:

require(PREDAsampledata)

data(ExpressionSetRCC)

GEstatisticsForPREDA<-statisticsForPREDAfromEset(
ExpressionSetRCC, statisticType="tstatistic",
referenceGroupLabel="normal", classVector=sampleinfo[,"Class"])


## End(Not run)
```

StatisticsForPREDAFromfile

Function to create a StatisticsForPREDA objet from a txt file

---

Description

Function to create a StatisticsForPREDA objet from a txt file

Usage

```
StatisticsForPREDAFromfile(file, ids_column = NULL,
statistic_columns = NULL, analysesNames = NULL, testedTail =
c("upper", "lower", "both"), ...)
```

Arguments

file	Path to the input txt file containing statistics on genomics data
ids_column	Specify the column from the input txt file with gene (or other genomic features) ids. Can be specified using column index (numeric) or column name (character).
statistic_columns	Specify the column (or columns) from the input txt file with gsta.enomic data statistics that will be included in the statisticsForPREDA object. Can be specified using column index (numeric) or column name (character). If NULL (default), all columns excluding ids_column will be considered as input statistics

analysesNames	Names (labels) to be associated to each input statistic. If NULL the column names for statistics_columns will be used.
testedTail	Specify what tail of the distribution will be tested for significantly extreme values in PREDA analysis. Possible values are "both", "upper" or "lower".
...	any other parameter for read.table function that could be useful for parsing the input file, such as "sep", "quote", "header", "na.strings" and other parameters.

Details

A txt file is parsed and a statisticsForPREDA object is built using contained data.

Value

A statisticsForPREDA object

Author(s)

Francesco Ferrari

See Also

StatisticsForPREDA

Examples

```
  ## Not run:
require(PREDAsampledata)

CNdataPath <- system.file("sampledata", "CopyNumber", package =
"PREDAsampledata")

CNdataFile <- file.path(CNdataPath , "CNAG_data_PREDA.txt")

CNannotationFile <- file.path(CNdataPath , "SNPAnnot100k.csv")

CNStatisticsForPREDA<-StatisticsForPREDAFromfile(file=CNdataFile,
ids_column="AffymetrixSNPsID", testedTail="both", sep="\t",
header=TRUE)


## End(Not run)
```

Index

∗ **classes**
DataForPREDA-class, 6
GenomicAnnotations-class, 14
GenomicAnnotationsForPREDA-class, 20
GenomicRegions-class, 31
PREDADataAndResults-class, 45
PREDAResults-class, 48
StatisticsForPREDA-class, 62

analysesNames, 4, *63*
analysesNames,PREDAResults-method *(*analysesNames*)*, 4
analysesNames,StatisticsForPREDA-method *(*analysesNames*)*, 4

computeDatasetSignature, 5
computeDatasetSignature,GenomicAnnotationsForPREDA-method *(*computeDatasetSignature*)*, 5

DataForPREDA, *46, 47*, *57*, *60*, *63*
DataForPREDA-class, 6
DataForPREDA2dataframe, *8*, 9
DataForPREDA2dataframe,DataForPREDA-method *(*DataForPREDA2dataframe*)*, 9
DataForPREDA2GenomicAnnotationsForPREDA, *8*, 9
DataForPREDA2GenomicAnnotationsForPREDA,DataForPREDA-method *(*DataForPREDA2GenomicAnnotationsForPREDA*)*, 9
DataForPREDA2StatisticsForPREDA, *8*, 10
DataForPREDA2StatisticsForPREDA,DataForPREDA-method *(*DataForPREDA2StatisticsForPREDA*)*, 10
DataForPREDAMedianCenter, 10
DataForPREDAMedianCenter,DataForPREDA-method *(*DataForPREDAMedianCenter*)*, 10

eset2GenomicAnnotations, 11
eset2GenomicAnnotations,ExpressionSet-method *(*eset2GenomicAnnotations*)*, 11
ExpressionSet, *62*

genomePlot, 12, *22*
genomePlot,GenomicAnnotationsForPREDA-method *(*genomePlot*)*, 12
GenomicAnnotations, *7, 8*, *11*, *19*, *21, 22*, *27–30*, *46, 47*, *49*, *50*
GenomicAnnotations-class, 14
GenomicAnnotations2dataframe, *15*, 15, *22*
GenomicAnnotations2dataframe,GenomicAnnotations-method *(*GenomicAnnotations2dataframe*)*, 15
GenomicAnnotations2dataframe,GenomicAnnotationsForPREDA-me *(*GenomicAnnotations2dataframe*)*, 15
GenomicAnnotations2GenomicAnnotationsForPREDA, *15*, 16, *25*
GenomicAnnotations2GenomicAnnotationsForPREDA,GenomicAnnot *(*GenomicAnnotations2GenomicAnnotationsForPREDA*)*, 16
GenomicAnnotations2reference_positions, *15*, 17
GenomicAnnotations2reference_positions,GenomicAnnotations- *(*GenomicAnnotations2reference_positions*)*, 17
GenomicAnnotationsExtract, *15*, 18
GenomicAnnotationsExtract,GenomicAnnotations-method *(*GenomicAnnotationsExtract*)*, 18
GenomicAnnotationsFilter_neg, *8*, *15*, 19, *22*
GenomicAnnotationsFilter_neg,DataForPREDA-method *(*GenomicAnnotationsFilter_neg*)*, 19
GenomicAnnotationsFilter_neg,GenomicAnnotations-method *(*GenomicAnnotationsFilter_neg*)*, 19
GenomicAnnotationsFilter_neg,GenomicAnnotationsForPREDA-me *(*GenomicAnnotationsFilter_neg*)*, 19

GenomicAnnotationsFilter_pos, *8*, *15*, 19, *22*
GenomicAnnotationsFilter_pos,DataForPREDA-method *(GenomicAnnotationsFilter_pos)*, 19
GenomicAnnotationsFilter_pos,GenomicAnnotations-method *(GenomicAnnotationsFilter_pos)*, 19
GenomicAnnotationsFilter_pos,GenomicAnnotationsForPREDA-method *(GenomicAnnotationsFilter_pos)*, 19
GenomicAnnotationsForPREDA, *7*, *8*, *13*, *16*, *25*, *46*, *47*, *49*, *50*
GenomicAnnotationsForPREDA-class, 20
GenomicAnnotationsForPREDA2dataframe, *22*, 22
GenomicAnnotationsForPREDA2dataframe,GenomicAnnotationsForPREDA-method *(GenomicAnnotationsForPREDA2dataframe)*, 22
GenomicAnnotationsForPREDA2GenomicAnnotations, *22*, 23
GenomicAnnotationsForPREDA2GenomicAnnotations,GenomicAnnotationsForPREDA-method *(GenomicAnnotationsForPREDA2GenomicAnnotations)*, 23
GenomicAnnotationsForPREDA2PREDAResults, *22*, 23
GenomicAnnotationsForPREDA2PREDAResults,GenomicAnnotationsForPREDA-method *(GenomicAnnotationsForPREDA2PREDAResults)*, 23
GenomicAnnotationsForPREDAFromfile, 24
GenomicAnnotationsFromdataframe, 26
GenomicAnnotationsFromfile, 27
GenomicAnnotationsFromLibrary, 29
GenomicAnnotationsSortAndCleanNA, *8*, *15*, *22*, 30, *33*, *47*, *50*
GenomicAnnotationsSortAndCleanNA,DataForPREDA-method *(GenomicAnnotationsSortAndCleanNA)*, 30
GenomicAnnotationsSortAndCleanNA,GenomicAnnotations-method *(GenomicAnnotationsSortAndCleanNA)*, 30
GenomicAnnotationsSortAndCleanNA,GenomicAnnotationsForPREDA-method *(GenomicAnnotationsSortAndCleanNA)*, 30
GenomicAnnotationsSortAndCleanNA,PREDADataAndResults-method *(GenomicAnnotationsSortAndCleanNA)*, 30
GenomicAnnotationsSortAndCleanNA,PREDAResults-method *(GenomicAnnotationsSortAndCleanNA)*, 30
GenomicRegions, *6*, *37*, *39*, *41*, *42*
GenomicRegions-class, 31
GenomicRegions2dataframe, 33
GenomicRegionsAnnotate, *15*, 34
GenomicRegionsAnnotate,GenomicRegions,GenomicAnnotations-method *(GenomicRegionsAnnotate)*, 34
GenomicRegionsChrNumber, 35
GenomicRegionsChrNumber,GenomicRegions-method *(GenomicRegionsChrNumber)*, 35
GenomicRegionsComparison, 36, *39*
GenomicRegionsComparison,GenomicRegions,GenomicRegions-method *(GenomicRegionsComparison)*, 36
GenomicRegionsCreateRegionsIds, 37
GenomicRegionsCreateRegionsIds,GenomicRegions-method *(GenomicRegionsCreateRegionsIds)*, 37
GenomicRegionsFilter_neg, 37
GenomicRegionsFilter_neg,GenomicRegions-method *(GenomicRegionsFilter_neg)*, 37
GenomicRegionsFilter_pos, 38
GenomicRegionsFilter_pos,GenomicRegions-method *(GenomicRegionsFilter_pos)*, 38
GenomicRegionsFindOverlap, *37*, 39
GenomicRegionsFromdataframe, 40
GenomicRegionsFromfile, 41
GenomicRegionsNumber, 43
GenomicRegionsNumber,GenomicRegions-method *(GenomicRegionsNumber)*, 43
GenomicRegionsSpan, 43
GenomicRegionsSpan,GenomicRegions-method *(GenomicRegionsSpan)*, 43
GenomicRegionsTotalSpan, 43
GenomicRegionsTotalSpan,GenomicRegions-method *(GenomicRegionsTotalSpan)*, 43
getStatisticByName, 44, *63*
getStatisticByName,StatisticsForPREDA-method *(getStatisticByName)*, 44

MergeStatisticAnnotations2DataForPREDA, 44

PREDA_main, 54
PREDAResults, *13*
PREDADataAndResults-class, 45
PREDADataAndResults2dataframe, *33*, *47*,

PREDADataAndResults2dataframe,PREDADataAndResults-method
(PREDADataAndResults2dataframe), 47
PREDAResults, *4*, *6*, *13*, *46, 47*
PREDAResults-class, 48
PREDAResults2dataframe, *50*, 50
PREDAResults2dataframe,PREDAResults-method
(PREDAResults2dataframe), 50
PREDAResults2GenomicRegions, *13*, *50*, 51
PREDAResults2GenomicRegions,PREDAResults-method
(PREDAResults2GenomicRegions), 51
PREDAResults2GenomicRegionsSingle, *50*, 52
PREDAResults2GenomicRegionsSingle,PREDAResults-method
(PREDAResults2GenomicRegionsSingle), 52
PREDAResults2PREDADataAndResults, *50*, 53
PREDAResults2PREDADataAndResults,PREDAResults-method
(PREDAResults2PREDADataAndResults), 53
PREDAResultsGetObservedFlags, *50*, 53
PREDAResultsGetObservedFlags,PREDAResults-method
(PREDAResultsGetObservedFlags), 53
preprocessingGE, 56, *60*, *62*

SODEGIR_GEstatistics, 61
SODEGIR_GEstatistics,ExpressionSet-method
(SODEGIR_GEstatistics), 61
SODEGIRpreprocessingGE, 58, *62*
StatisticsForPREDA, *4*, *7, 8*, *46, 47*, *66*, *68, 69*
StatisticsForPREDA-class, 62
StatisticsForPREDA2dataframe, *63*, 64
StatisticsForPREDA2dataframe,StatisticsForPREDA-method
(StatisticsForPREDA2dataframe), 64
StatisticsForPREDAFilterColumns_neg, *8*, *63*, 64
StatisticsForPREDAFilterColumns_neg,DataForPREDA-method
(StatisticsForPREDAFilterColumns_neg), 64
StatisticsForPREDAFilterColumns_neg,StatisticsForPREDA-method
(StatisticsForPREDAFilterColumns_neg), 64
StatisticsForPREDAFilterColumns_pos, *8*, *63*, 65
StatisticsForPREDAFilterColumns_pos,DataForPREDA-method
(StatisticsForPREDAFilterColumns_pos), 65
StatisticsForPREDAFilterColumns_pos,StatisticsForPREDA-method
(StatisticsForPREDAFilterColumns_pos), 65
StatisticsForPREDAFromdataframe, 65
statisticsForPREDAfromEset, 67
statisticsForPREDAfromEset,ExpressionSet-method
(statisticsForPREDAfromEset), 67
StatisticsForPREDAFromfile, 68