Package 'SNPhood'

April 16, 2019

Title SNPhood: Investigate, quantify and visualise the epigenomic neighbourhood of SNPs using NGS data

Version 1.12.0

Author Christian Arnold [aut, cre], Pooja Bhat [aut], Judith Zaugg [aut]

Maintainer Christian Arnold <christian.arnold@embl.de>

Description To date, thousands of single nucleotide polymorphisms (SNPs) have been found to be associated with complex traits and diseases. However, the vast majority of these disease-associated SNPs lie in the non-coding part of the genome, and are likely to affect regulatory elements, such as enhancers and promoters, rather than function of a protein. Thus, to understand the molecular mechanisms underlying genetic traits and diseases, it becomes increasingly important to study the effect of a SNP on nearby molecular traits such as chromatin environment or transcription factor (TF) binding. Towards this aim, we developed SNPhood, a user-friendly *Bioconductor* R package to investigate and visualize the local neighborhood of a set of SNPs of interest for NGS data such as chromatin marks or transcription factor binding sites from ChIP-Seq or RNA-Seq experiments. SNPhood comprises a set of easy-to-use functions to extract, normalize and summarize reads for a genomic region, perform various data quality checks, normalize read counts using additional input files, and to cluster and visualize the regions according to the binding pattern. The regions around each SNP can be binned in a user-defined fashion to allow for analysis of very broad patterns as well as a detailed investigation of specific binding shapes. Furthermore, SNPhood supports the integration with genotype information to investigate and visualize genotype-specific binding patterns. Finally, SNPhood can be employed for determining, investigating, and visualizing allele-specific binding patterns around the SNPs of interest.

Imports DESeq2, cluster, ggplot2, lattice, GenomeInfoDb, BiocParallel, VariantAnnotation, BiocGenerics, IRanges, methods, SummarizedExperiment, RColorBrewer, Biostrings, grDevices, gridExtra, stats, grid, utils, reshape2, scales, S4Vectors

Depends R (>= 3.1), GenomicRanges, Rsamtools, data.table, checkmate

Suggests BiocStyle, knitr, pryr, rmarkdown, SNPhoodData, corrplot

VignetteBuilder knitr

biocViews Software

License LGPL (>= 3)

LazyData true

URL https://bioconductor.org/packages/SNPhood

BugReports christian.arnold@embl.de

RoxygenNote 5.0.1

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

git_branch RELEASE_3_8

git_last_commit f6e5f8e

git_last_commit_date 2018-10-30

Date/Publication 2019-04-15

R topics documented:

`analyzeSNPhood` *Main function of* SNPhood

Description

`analyzeSNPhood` is the main function of the `SNPhood` package. All results, parameters and metadata are stored in an object of class `SNPhood`.

Usage

```
analyzeSNPhood(par.l, files.df, onlyPrepareForDatasetCorrelation = FALSE,
  verbose = TRUE)
```

Arguments

`par.l`	Named list. Named list with all required parameter names and their respective values, which should be generated via the helper function `getDefaultParameterList`. Note that all supported parameters must be defined in the list, as obtained by the function `getDefaultParameterList` . See also ?getDefaultParameterList for details.
`files.df`	Data frame with at least the column "signal" specifying the absolute paths to the BAM files that will be processed. Optionally, further columns can be added. Supported are "input", "individual" and "genotype". See the Vignette for further details. The data frame can either be created manually or via the helper function `collectFiles`.
`onlyPrepareForDatasetCorrelation`	Logical(1). Default FALSE. If set to TRUE, only steps necessary to analyze the correlation among datasets with respect to their read counts are calculated, which is less thsan time-consuming than running the full pipeline. This is a quality control step to identify outlier datasets that show artefacts and that should therefore be removed from the analysis. If set to FALSE (the default), the full pipeline is executed. In both cases, the function `plotAndCalculateCorrelationDatasets` can be executed afterwards.
`verbose`	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Details

If you already have BAM files in objects of class `BamFile` or `BamFileList`, see the function `collectFiles` for how to seemlessly integrate them into the `SNPhood` framework.

In addition, see the vignettes for more details.

Value

Object of class SNPhood. See the class description (?"SNPhood-class", or click the link) for details.

Examples

```
## For the following example, see also the workflow vignette!
library(SNPhoodData)
# get a list of files to process
dataDir = system.file("extdata", package = "SNPhoodData")
files.df = collectFiles(patternFiles = paste0(dataDir,"/*.bam"))
files.df$individual = c("GM10847", "GM10847", "GM12890", "GM12890")
fileUserRegions = list.files(pattern = "*.txt",dataDir, full.names = TRUE)
par.l = getDefaultParameterList(path_userRegions = fileUserRegions)
par.l$poolDatasets = TRUE
# Run the main function with the full pipeline
SNPhood.o = analyzeSNPhood (par.l, files.df)
```

---

annotation,SNPhood-method

Retrieve the annotation of a SNPhood *object.*

Description

Specific elements within the annotation slot may also be extracted by using the `elements` parameter.

Usage

```
## S4 method for signature 'SNPhood'
annotation(object, elements = NULL, ...)
```

Arguments

`object`	Object of class SNPhood
`elements`	Character. The name of the element(s) in the annotation slot to be extracted. If set to NULL, the full annotation slot is returned.
`...`	not supported

Value

If only a single value for `elements` is provided, the element is returned directly. If multiple values are provided, a named list with the requested elements is returned.

Examples

```
data(SNPhood.o, package="SNPhood")
annotation(SNPhood.o)
annotation(SNPhood.o, elements = "regions")
annotation(SNPhood.o, elements = c("regions", "bins"))
```

annotationBins *Get the annotation(names) of the bins in a* SNPhood *object.*

Description

Return the names of the Bins that are defined in the SNPhood object.

Usage

```
annotationBins(SNPhood.o, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Character vector. Names of the bins that are defined in the SNPhood object.

Examples

```
data(SNPhood.o, package="SNPhood")
annotationReadGroups(SNPhood.o)
```

annotationBins2 *Get the annotation(names) of bins in a* SNPhood *object.*

Description

annotationBins2 is a helper function that returns annotation of the bins that are defined in the SNPhood object.

Usage

```
annotationBins2(SNPhood.o, regions = NULL, fullAnnotation = FALSE,
  verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
regions	Integer or character. Default NULL. A subset of the SNP regions for which annotation is needed. Either the row numbers or the rownames(IDs) of the SNP regions are supported.
fullAnnotation	Logical(1). Should the full annotation(as a data.frame) be returned or only the annotation of the bins(as a character vector)?
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

If `fullAnnotation` is set to `TRUE`, a data.frame with the full annotation of the bins for the(subset of) SNP regions is returned. Otherwise, a character vector with only the annotation of the bins is returned.

Warning

The number of returned bins can easily be very large, in the order of millions. Be careful because the memory consumption due the resulting object may increase considerably. Reduce memory requirements by returning only a subset of SNP regions

Examples

```
data(SNPhood.o, package="SNPhood")
annotation.df  = annotationBins2(SNPhood.o, regions = 1:10, fullAnnotation = TRUE)
annotation.vec = annotationBins2(SNPhood.o, regions = 1:10, fullAnnotation = FALSE)
```

---

annotationDatasets *Get the annotation(names) of the datasets in a* SNPhood *object.*

Description

Return the names of the datasets/individuals that are defined in the SNPhood object.

Usage

```
annotationDatasets(SNPhood.o, verbose = FALSE)
```

Arguments

`SNPhood.o`	Object of class SNPhood
`verbose`	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Character vector. Names of the datasets/individuals that are defined in the SNPhood object.

Examples

```
data(SNPhood.o, package="SNPhood")
annotationDatasets(SNPhood.o)
```

annotationReadGroups *Get the annotation(names) of the read groups in a* SNPhood *object.*

Description

Return the names of the read groups that are defined in the SNPhood object.

Usage

```
annotationReadGroups(SNPhood.o, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Character vector. Names of the read groups that are defined in the SNPhood object.

Examples

```
data(SNPhood.o, package="SNPhood")
annotationReadGroups(SNPhood.o)
```

annotationRegions *Get the annotation of SNP regions for a* SNPhood *object.*

Description

Return the annotation of the SNP regions that are defined in the SNPhood object.

Usage

```
annotationRegions(SNPhood.o, asGRangesObj = FALSE, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
asGRangesObj	Logical(1). Default FALSE. Should the full annotation be returned (as GRanges object) or only the annotation of the SNP regions (as character vector)?
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

If asGRangesObj is set to TRUE, a GRanges object is returned. Otherwise, a character vector with the currently stored SNP annotation is returned.

Examples

```
data(SNPhood.o, package="SNPhood")
IDs.vec = annotationRegions(SNPhood.o, asGRangesObj = FALSE)
IDs.gr  = annotationRegions(SNPhood.o, asGRangesObj = TRUE)
```

`associateGenotypes` *Associate genotypes with user regions from a* SNPhood *object.*

Description

The function `associateGenotypes` associates genotypes with SNP regions as defined in a SNPhood object. It is possible to assign genotypes only for a subset of datasets as defined in a SNPhood object. To avoid any ambiguities, a 1:1 for genotype and dataset mapping must be given (ses below).

Usage

```
associateGenotypes(SNPhood.o, genotypeMapping, verbose = TRUE)
```

Arguments

`SNPhood.o`	Object of class SNPhood
`genotypeMapping`	Data frame. A data frame that establishes the mapping between datasets in the object and the corresponding genotype file and column names. See the examples. must be provided. See the Vignette for a more detailed description of the supported file format.
`verbose`	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Object of class SNPhood with the genotype information added to the slot `annotation`, element genotype. You may retrieve it via the accessor function `annotation`.

Examples

```
data(SNPhood.o, package="SNPhood")
fileGenotypes = list.files(pattern = "*genotypes*",system.file("extdata", package = "SNPhoodData"), full.nam
mapping = data.frame(samples = annotationDatasets(SNPhood.o), genotypeFile = rep(fileGenotypes, 2), sampleNa
SNPhood.o = associateGenotypes(SNPhood.o, mapping)
```

changeObjectIntegrityChecking

Disable object integrity checking for a SNPhood *object.*

Description

The function changeObjectIntegrityChecking disables object integrity checking for a *SNPhood* object. This might be desired for large objects when the integrity test takes too much time. Note, however, that disabling these checks is not recommended.

Usage

```
changeObjectIntegrityChecking(SNPhood.o, disable = FALSE, verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
disable	Logical(1). Default FALSE. Disable the object integrity checking?
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Object of class SNPhood with object integrity checking disabled.

Examples

```
data(SNPhood.o, package="SNPhood")
SNPhood.o = changeObjectIntegrityChecking(SNPhood.o, disable = TRUE)
```

collectFiles

Helper function to generate a data frame that can be used as input for the function analyzeSNPhood

Description

collectFiles creates a data frame that can be used as input for the function analyzeSNPhood. The resulting data frame contains information about files that will be processed (column signal) and, optionally, corresponding input files for normalization (column input) and labels to combine datasets to meta-datasets (column individual).

Usage

```
collectFiles(patternFiles, recursive = FALSE, ignoreCase = TRUE,
  inputFiles = NA, individualID = NA, genotypeMapping = NA,
  verbose = TRUE)
```

Arguments

patternFiles	Character. If vector of length 1, absolute path to one or multiple BAM files that should be processed. Wildcards ("*") are allowed (examples are *CTCF* or *.bam, see also examples). If vector of length > 1, each element must specify the absolute path to a BAM file, with no wildcards being allowed. See also the note above concerning the integration of BamFile or BamFileList objects. For more details, see the examples and the vignette.
recursive	Logical(1). Default FALSE. Should the search for BAM files within the directory be performed recursively? If set to TRUE, all files matching the pattern within the specified directory and all of its subdirectories will be added. If set to FALSE, only files within the specified directory but not any subdirectories will be used.
ignoreCase	Logical(1). Default TRUE. Should the specified pattern be case sensitive?
inputFiles	Character. Default NULL. Input files that should be used as a control for normalization. Supported values are NA (no input normalization), a single character specifying one or multiple input files (comma-separated, see examples) that should be used for all processed files, or a character vector of the same length as the number of files that will be processed. Set to NULL if you want to add the files later manually in the data frame (see vignette).
individualID	Character. Default NULL. Name of the individual IDs. Only relevant if datasets should be pooled.
genotypeMapping	Character. Default NULL. Path to the corresponding genotype file in VCF format, followed by a colon and the name of the column in the VCF file. Genotypes can also be integrated later using the function associateGenotypes
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Details

Note that if you already have an object of class BamFile or BamFileList, this can easily be integrated into the SNPhood framework by using the path function to specify the value of the parameter patternFiles, see the examples below.

Value

a data frame with the three columns signal, input and individual that can be used as input for the function analyzeSNPhood.

See Also

analyzeSNPhood

Examples

```
## For brevity, only exemplary filenames are given in the following.
## Note that in reality, absolute paths should be provided.
## First some examples using specific files rather than files that
## match a pattern in a particular directory

## Load SNPhoodData library
library(SNPhoodData)
```

```
files.df = collectFiles(patternFiles = paste0(system.file("extdata", package = "SNPhoodData"),"/*.bam"))

## If you already have BAM files in objects of class \code{\linkS4class{BamFile}} or \code{\linkS4class{BamFi
## you may use the following code snippet:
files = list.files(pattern = "*.bam$",system.file("extdata", package = "SNPhoodData"),full.names = TRUE)
BamFile.o = BamFile(files[1])
BamFiles.o = BamFileList(files)
files.df = collectFiles(patternFiles = path(BamFile.o))
files.df = collectFiles(patternFiles = path(BamFiles.o))
```

`convertToAllelicFractions` *Convert read counts across read groups to relative fractions from a* SNPhood *object.*

Description

`convertToAllelicFractions` convert read counts across read groups to their relative fractions among all read groups (all read counts will be between 0 and 1, with 1 for a particular read group depicting that all reads from this particular position originate from the one read group) Affected slots are `readCountsUnbinned` and `readCountsBinned`. It is recommended to save the resulting SNPhood object with a new name because it is not possible to go back from fractions to read counts at a later point.

Usage

```
convertToAllelicFractions(SNPhood.o, roundDigits = 2, setNaNToZero = FALSE,
  verbose = TRUE)
```

Arguments

`SNPhood.o`	Object of class SNPhood
`roundDigits`	Numeric(1). Default 2. Number of digits after the decimal place when converting read counts to fractions
`setNaNToZero`	Logical(1). Default FALSE. Should NaN (not a number) be converted to 0? NaN may result from individual regions or bins with no reads across all read groups due to a division by zero.
`verbose`	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

an object of class SNPhood with read counts across read groups (both for the slots readCountsUnbinned and readCountsBinned) replaced by their respective relative fractions. Otherwise identical to the input SNPhood object.

See Also

`deleteReadGroups`

Examples

```
data(SNPhood.o, package="SNPhood")
SNPhood_allelicFractions.o = convertToAllelicFractions(SNPhood.o)

# Convert all NaN to 0 for subsequent analyses
SNPhood_allelicFractions.o = convertToAllelicFractions(SNPhood.o, setNaNToZero = TRUE)
```

---

counts,SNPhood-method *Extract count data from a SNPhood object.*

Description

counts extracts count data from a SNPhood object. The full count data or only a subset can be extracted by settings the parameters type, readGroup and dataset accordingly. Either the count data for the unbinned or binned SNP regions can be extracted.

Usage

```
## S4 method for signature 'SNPhood'
counts(object, type = "binned", readGroup = NULL,
  dataset = NULL, ...)
```

Arguments

object	Object of class SNPhood
type	Character(1). Default "binned". Either "binned" or "unbinned" to extract counts after or before binning the SNP regions, respectively.
readGroup	Character(1). Default NULL. Read group that should be plotted, specified by its name as obtained by the function annotationReadGroups). If only one read group is defined in the object, this may also be NULL for user conveniance.
dataset	Numeric(1) or Character(1). Single dataset that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or its annotation (name must appear in the dataset names as obtained via the function annotationDatasets).
...	not used

Value

A named nested list with the requested count data, organized after read group and dataset.

See Also

SNPhood, enrichment

Examples

```
data(SNPhood.o, package="SNPhood")
str(counts(SNPhood.o))
str(counts(SNPhood.o, readGroup = "paternal", dataset = 1))
str(counts(SNPhood.o, readGroup = c("maternal", "paternal"), dataset = 1))
```

deleteDatasets *Delete a particular set of datasets from a* SNPhood *object.*

Description

deleteDatasets deletes a particular set of datasets from a SNPhood object. Removal is irreversible. It is therefore recommended to save the resulting SNPhood object with a new name because the deleted datasets cannot be recovered.

Usage

```
deleteDatasets(SNPhood.o, datasets = NULL, verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
datasets	Numeric or Character or NULL. Default NULL. Datasets that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or their annotation (name must appear in the dataset names as obtained via the function annotationDatasets). If set to NULL, all datasets will be considered.
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

an object of class SNPhood with the requested datasets removed from all slots.

See Also

deleteRegions, deleteReadGroups

Examples

```
data(SNPhood.o, package="SNPhood")
SNPhood_mod.o = deleteDatasets(SNPhood.o, c(1,2))
```

deleteReadGroups *Delete a particular set of read groups.*

Description

deleteReadGroups deletes a particular set of read groups from a *SNPhood* object. Removal is irreversible. It is therefore recommended to save the resulting SNPhood object with a new name.

Usage

```
deleteReadGroups(SNPhood.o, readGroups = NULL, verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
readGroups	Character or NULL. Default NULL. Read groups that should be plotted, specified by their name as obtained by the function annotationReadGroups). If set to NULL, all read groups will be considered.
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

an object of class SNPhood with read counts across read groups (both for the slots readCountsUnbinned and readCountsBinned) replaced by their respective relative fractions. Otherwise identical to the input SNPhood object.

See Also

deleteDatasets, deleteRegions

Examples

```
data(SNPhood.o, package="SNPhood")
SNPhood_allelicFractions.o = deleteReadGroups(SNPhood.o, "ambiguous")
```

---

deleteRegions *Delete a set of user regions from a* SNPhood *object.*

Description

deleteRegions deletes a set of user regions. Removal is irreversible. It is therefore recommended to save the resulting SNPhood object with a new name.

Usage

```
deleteRegions(SNPhood.o, regions, verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
regions	Numeric or Character or NULL. Default NULL. Regions that should be plotted, either specified as integer (such as 1, value must be between 1 and the total number of regions as defined in the object) or their annotation (name must appear in the region names as obtained via the function annotationRegions). If set to NULL, all regions will be considered.
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

an object of class SNPhood with the requested regions being deleted.

Warning

Execution of this function resets the slot additionalResults and all of its results (e.g, allelic bias analysis). The reason for this is that all results stored in this slot are affected by the deletion of regions

See Also

deleteDatasets, deleteReadGroups

Examples

```
data(SNPhood.o, package="SNPhood")
# Delete the first 10 regions
SNPhood_mod.o = deleteRegions(SNPhood.o, c(1:10))

# Delete regions by their annotation
SNPhood_mod.o = deleteRegions(SNPhood.o, c("rs2822405", "rs467140"))
```

---

enrichment *Extract enrichment data from an object.*

Description

Extract enrichment data from an object.

enrichment extracts enrichment data from a SNPhood object. The full count data or only a subset can be extracted by settings the parameters type, readGroup and dataset accordingly. Principally, either the count data for the unbinned or binned SNP regions can be extracted.

Usage

```
enrichment(object, ...)

## S4 method for signature 'SNPhood'
enrichment(object, readGroup = NULL, dataset = NULL,
  ...)
```

Arguments

object	An object containing enrichment information.
...	Additional arguments, for use in specific methods.
readGroup	Character(1). Default NULL. Read group that should be plotted, specified by its name as obtained by the function annotationReadGroups). If only one read group is defined in the object, this may also be NULL for user conveniance.
dataset	Numeric(1) or Character(1). Single dataset that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or its annotation (name must appear in the dataset names as obtained via the function annotationDatasets).

Value

Enrichment of the object or the objects components.

Named list with the requested enrichment matrices from the SNPhood object, organized by read group and dataset

See Also

counts

Examples

```
data(SNPhood.o, package="SNPhood")
str(enrichment(SNPhood.o), list.len=5)
```

---

getDefaultParameterList

Helper function to generate a default parameter list as input for the function analyzeSNPhood

---

Description

getDefaultParameterList generates a default parameter list that can be used as input for the function analyzeSNPhood. The path to the user regions file can optionally be provided as an argument to the function. See the examples for further details. Before running the function analyzeSNPhood, carefully check that the default parameters are suitable for the analysis.

Usage

```
getDefaultParameterList(path_userRegions = NULL, isPairedEndData = TRUE)
```

Arguments

path_userRegions
: Character(1). Specify the value of the parameter path_userRegions (absolute path to the user regions file, see the Vignette for details).

isPairedEndData
: Logical(1). Default TRUE. Are the data paired-end (TRUE) or single-end (FALSE)?

Value

a named list with default values for the currently supported parameters that can be used as input for the function analyzeSNPhood:

- readFlag_isPaired: Logical(1), TRUE for paired-end data, NA for single-end
- readFlag_isProperPair: Logical(1), TRUE
- readFlag_isUnmappedQuery: Logical(1), FALSE
- readFlag_hasUnmappedMate: Logical(1), FALSE
- readFlag_isMinusStrand: Logical(1), NA
- readFlag_isMateMinusStrand: Logical(1), NA

- readFlag_isFirstMateRead: Logical(1), NA
- readFlag_isSecondMateRead: Logical(1), NA
- readFlag_isNotPrimaryRead: Logical(1), FALSE
- readFlag_isNotPassingQualityControls: Logical(1), FALSE
- readFlag_isDuplicate: Logical(1), FALSE
- readFlag_reverseComplement: Logical(1), FALSE
- readFlag_simpleCigar: Logical(1), TRUE
- path_userRegions: Character(1), as given by the function argument path_userRegions
- zeroBasedCoordinates: Logical(1), FALSE
- regionSize: Integer(1), 500
- binSize: Integer(1), 50
- readGroupSpecific: Logical(1), TRUE
- strand: Character(1), "both"
- startOpen: Logical(1), FALSE
- endOpen: Logical(1), FALSE
- headerLine: Logical(1), FALSE
- linesToParse: Integer(1), -1
- lastBinTreatment: Character(1), "delete"
- assemblyVersion: Character(1), "hg19"
- nCores: Integer(1), 1
- keepAllReadCounts: Logical(1), FALSE
- normByInput: Logical(1), FALSE
- normAmongEachOther: Logical(1), TRUE
- poolDatasets: Logical(1), FALSE

For reasons of reduced redundancy, a detailed description of the parameters can be found at the end of the main vignette in SNPhood (`browseVignettes("SNPhood")`).

See Also

`analyzeSNPhood`

Examples

```
## Only one parameter can, optionally, be specified when calling the function
par.l = getDefaultParameterList(path_userRegions = "path/to/regions", isPairedEndData = TRUE)
## If the file is not specified, you need to change it
## before you can execute the function \code{\link{analyzeSNPhood}}
par.l = getDefaultParameterList(isPairedEndData = TRUE)
par.l$path_userRegions = "path/to/regions"
```

mergeReadGroups *Merges the counts of all read groups for a* SNPhood *object*

Description

mergeReadGroups merges the counts of all read groups for a SNPhood object. This function can only be executed if more than one read group is defined in the object and if read counts have not been converted into allelic fractions. Also carefully note the warning below.

Usage

```
mergeReadGroups(SNPhood.o, summaryFunction = "sum", verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
summaryFunction	Character(1). Default "sum". Either "sum" or "mean". How should the read counts from different read groups be summarized. If set to "sum", all counts are summed up, which yields values that are identical as running the main analysis non-allele-specifically. If set to "mean", the mean value across all read groups is calculated.
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

A modified SNPhood object with only one read group "allReadGroups", with all occurences of the original read groups replaced by "allReadGroups". For object consistency, as mentioned in the warning below, some results from analyses depending on read groups are removed completely.

Warning

Merging read groups is irreversible. This transformation cannot be undone. It might therefore be advisable to save the resulting object in a new variable as shown in the examples.

Results from the allelic bias test and clustering results will also be removed to keep the object consistent.

Examples

```
data(SNPhood.o, package="SNPhood")
nReadGroups(SNPhood.o)
SNPhood_merged.o = mergeReadGroups(SNPhood.o)
nReadGroups(SNPhood.o)
```

nBins — *Get the number of bins for a* SNPhood *object.*

Description

Return the number of bins that are defined in the SNPhood object.

Usage

```
nBins(SNPhood.o, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Integer. Number of bins that are defined in the SNPhood object

Examples

```
data(SNPhood.o, package="SNPhood")
nBins(SNPhood.o)
```

nDatasets — *Get the number of datasets for a* SNPhood *object.*

Description

Return the number of datasets that are defined in the SNPhood object.

Usage

```
nDatasets(SNPhood.o, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Integer. Number of datasets that are defined in the SNPhood object

Examples

```
data(SNPhood.o, package="SNPhood")
nDatasets(SNPhood.o)
```

nReadGroups *Get the number of read groups for a* SNPhood *object.*

Description

Return the number of read groups that are defined in the SNPhood object.

Usage

```
nReadGroups(SNPhood.o, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Integer. Number of read groups that are defined in the SNPhood object

Examples

```
data(SNPhood.o, package="SNPhood")
nReadGroups(SNPhood.o)
```

nRegions *Get the number of SNP regions for a* SNPhood *object.*

Description

nRegions is a helper function that returns the number of SNP regions that are defined in the SNPhood object.

Usage

```
nRegions(SNPhood.o, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Integer. Number of SNP regions that are defined in the SNPhood object

Examples

```
data(SNPhood.o, package="SNPhood")
nRegions(SNPhood.o)
```

parameters *Retrieve the parameters of an object.*

Description

Retrieve the parameters of an object.

Retrieve the parameters of a SNPhood object.

Usage

```
parameters(object, ...)

## S4 method for signature 'SNPhood'
parameters(object, ...)
```

Arguments

object	An object containing parameters with which it was created.
...	Additional arguments, for use in specific methods.

Value

A named list with all parameters and its current values of the SNPhood object.

Examples

```
data(SNPhood.o, package="SNPhood")
parameters(SNPhood.o)
```

plotAllelicBiasResults *Graphically summarize the results of the allelic bias analysis for a specific dataset and region.*

Description

plotAllelicBiasResults graphically summarizes the results of the allelic bias analysis for a specific dataset and region. Three plots are generated, each of which focuses on a different aspect of the allelic bias analysis across the selected user region.

Usage

```
plotAllelicBiasResults(SNPhood.o, dataset = 1, region = 1,
  signThreshold = 0.05, readGroupColors = NULL, fileToPlot = NULL,
  verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
dataset	Numeric(1) or Character(1). Single dataset that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or its annotation (name must appear in the dataset names as obtained via the function annotationDatasets).
region	Numeric(1) or Character(1). Single region that should be plotted, either specified as integer (such as 1, value must be between 1 and the total number of region as defined in the object) or its annotation (name must appear in the region names as obtained via the function annotationRegions).
signThreshold	Numeric(1). Default 0.05. The significance threshold (such as p-value or FDR threshold). Must be between 0 and 1. If the parameter belongs to a plotting function, a horizontal line is drawn at the chosen value. For the allelic bias summary plots, p-values below this threshold and the corresponding allelic fractions are highlighted.
readGroupColors	Character or NULL. Default NULL. Colors of the read groups that appear in the plot. If set to NULL, colors will be set automatically. The length of the vector must equal the **total** number of read groups that are defined in the SNPhood object.
fileToPlot	Character(1) or NULL. Default NULL. Filename of the PDF file for the output plots. If set to NULL, plots will be plotted to the currently active device.
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Details

The first plot shows the estimates of the allelic fraction, along with confidence intervals for the estimate according to the parameters chosen when the function testForAllelicBias was called. Fraction estimates for which the corresponding p-values are deemed significant according to the value of the parameter signThreshold are highlighted (see also the legend). At 0.5, the estimated allelic fraction if there was no allelic bias, a horizontal line is drawn.

The second plot shows the p-values (-log 10 transformed, so that smaller p-values have higher transformed values). In analogy to the estimates of the allelic fraction, significant p-values are highlighted. The -log10 transformed significance threshold (according to the parameter signThreshold) appears as a horizontal line.

Finally, the third plot shows the distribution of the read counts across all read groups. In addition, the genotype distribution for each read group is given (see the Vignette for details). This helps to identify allelic biases based on genotype differences among read groups.

Value

the generated **ggplot2** plot(s) as list for further processing. May contain multiple plots, depending on the function. The plot(s) can then be plotted individually or modified arbitrarily as the user wants. For example, if multiple plots are returned and the plots have been saved in a variable called plots.l, simply type plots.l[[1]] to view the first plot.

Examples

```
data(SNPhood.o, package="SNPhood")
```

```
SNPhood.o = testForAllelicBiases(SNPhood.o, readGroups = c("maternal", "paternal"))
# Leave all parameters with their standard values
plots = plotAllelicBiasResults(SNPhood.o)

# Change the colors
plots = plotAllelicBiasResults(SNPhood.o, readGroupColors = c("blue", "red", "gray"))

# Alter the significance threshold
plots = plotAllelicBiasResults(SNPhood.o, signThreshold = 0.01)
```

`plotAllelicBiasResultsOverview` *Visualize the results of the allelic bias analysis across regions or a user-defined genomic range*

Description

`plotBinCounts` visualizes the results of the allelic bias analysis across regions or a user-defined genomic range. Note that only the results of a particular chromosome can be visualized. It is therefore only possible if the regions to be visualized are located on one particular chromosome; otherwise, an error is thrown.

Usage

```
plotAllelicBiasResultsOverview(SNPhood.o, regions = 1, datasets = NULL,
  plotChr = NULL, plotStartPos = NULL, plotEndPos = NULL, ylim = NULL,
  plotRegionBoundaries = FALSE, plotRegionLabels = FALSE,
  signThreshold = 0.05, pValueSummary = "min", maxWidthLabels = NULL,
  colorPalette = "Set1", sizePoints = 4, printPlot = TRUE,
  fileToPlot = NULL, verbose = FALSE)
```

Arguments

`SNPhood.o`	Object of class `SNPhood`
`regions`	Numeric or Character or NULL. Default NULL. Regions that should be plotted, either specified as integer (such as 1, value must be between 1 and the total number of regions as defined in the object) or their annotation (name must appear in the region names as obtained via the function `annotationRegions`). If set to NULL, all regions will be considered.
`datasets`	Numeric or Character or NULL. Default NULL. Datasets that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or their annotation (name must appear in the dataset names as obtained via the function `annotationDatasets`). If set to NULL, all datasets will be considered.
`plotChr`	Character(1) or NULL. Default NULL. The name of the chromosome for which the visualization should be done. Must be a valid chromosome name. If set to NULL, other parameters (such as`regions`) determine which genomic region should be plotted.

`plotStartPos` Character(1) or NULL. Default NULL. The start coordinates for which the visualization should be done. Must be a valid number with respect to the chromosome it refers to. If set to NULL and the parameter `plotChr` is not NULL, the start coordinates are set to 1.

`plotEndPos` Character(1) or NULL. Default NULL. The end coordinates for which the visualization should be done. Must be a valid number with respect to the chromosome it refers to. If set to NULL and the parameter `plotChr` is not NULL, the end coordinates are determined automatically and the full chromosome will be plotted.

`ylim` Numeric(2). Default NULL. Range of the y-axis, as specified by a minimum and a maximum value. See ?ylim for details.

`plotRegionBoundaries`
Logical(1). Default FALSE. Should the region boundaries be drawn in the plot? If set to TRUE, two vertical lines will be drawn for each region, corresponding to the region boundaries upstream and downstream of the SNP. This visual aid may help to judge the size of the regions and overlaps among regions.

`plotRegionLabels`
Logical(1). Should the annotation of the regions be drawn vertically below the x axis? If many regions are plotted, labels may overlap; however, for a few regions, this is usually not a problem.

`signThreshold` Numeric(1). Default 0.05. The significance threshold (such as p-value or FDR threshold). Must be between 0 and 1. If the parameter belongs to a plotting function, a horizontal line is drawn at the chosen value. For the allelic bias summary plots, p-values below this threshold and the corresponding allelic fractions are highlighted.

`pValueSummary` Character(1). Default "min". Either "min" or "median". If set to "min", for each region, the minimum p-value across all bins is displayed as a representative result for the region. This is in analogy to how the background caclulation for the FDR calculation works, see the vignette for details. If set to "median", the median p-value is calculated for each region and plotted. This may facilitate to identify regions for which a lot of bins have low p-values.

`maxWidthLabels` Numeric(1). Default NULL. Maximum width of the legend labels in number of characters. If the width of the legend labels are longer, they are shortened. Set to NULL to not shorten labels.

`colorPalette` Character(1). Default "Set1". Name of the palette from the `RColorBrewer` package from the qualitative palettes for the colors of the datasets that are plotted. Allowed palette names are "Accent", "Dark2", "Paired", "Pastel1", "Pastel2", "Set1", "Set2", and "Set3". Colors for the datasets are then determined automatically from the given palette name (from left to right, depending on the number of datasets to be plotted). The colors for the read groups within each datasets are based on the colors for the dataset, but with different saturation values.

`sizePoints` Numeric(1). Default 4. Size of the points that are drawn in the plot (if type is set to the default value of "p"). This parameter has no effect if type is set to "l".

`printPlot` Logical(1). Default TRUE. Should the plots be printed? Only relevant if `fileToPlot` is set to NULL; otherwise, the plots are always printed to the output file.

`fileToPlot` Character(1) or `NULL`. Default `NULL`. Filename of the PDF file for the output plots. If set to `NULL`, plots will be plotted to the currently active device.

`verbose` Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

the generated **ggplot2** plot(s) as list for further processing. May contain multiple plots, depending on the function. The plot(s) can then be plotted individually or modified arbitrarily as the user wants. For example, if multiple plots are returned and the plots have been saved in a variable called plots.l, simply type plots.l[[1]] to view the first plot.

Examples

```
data(SNPhood.o, package="SNPhood")

# Plot the allelic bias results for the first region using default values for all parameters
plots = plotAllelicBiasResultsOverview(SNPhood.o)

# Plot the allelic bias results for the full chr21
plots = plotAllelicBiasResultsOverview(SNPhood.o, regions = NULL, plotChr = "chr21")
```

---

plotAndCalculateCorrelationDatasets

Calculate and plot correlation of region read counts among pairs of input files.

Description

`plotAndCalculateCorrelationDatasets` calculates and plots the pairwise correlation of all pairs of input files with among each other. The main purpose is to identify artefacts with particular files that should subsequently be excluded. The correlation is based on the raw region read counts(i.e., before binning). The results of the correlation analysis are stored in the SNPhood object. If the `corrplot` package is available, it will be used to produce a nice visualization of the correlation matrix.

Usage

```
plotAndCalculateCorrelationDatasets(SNPhood.o, fileToPlot = NULL,
  corMeasure = "pearson", verbose = FALSE, ...)
```

Arguments

`SNPhood.o`	Object of class SNPhood
`fileToPlot`	Character(1) or `NULL`. Default `NULL`. Filename of the PDF file for the output plots. If set to `NULL`, plots will be plotted to the currently active device.
`corMeasure`	Character(1). Default "pearson". The correlation measure that should be used to compare between pairs of samples. Either `pearson`, `spearman`, or `kendall`.
`verbose`	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?
`...`	Additional arguments for the `corrplot.mixed` function from the `corrplot` package (if available).

Value

An object of type SNPhood, with the results of the correlation analysis stored in the slot "additionalResults". They can be retrieved via the helper function `results` for further investigation. The results consist of a named list with two elements: A correlation matrix of the region read counts across all input files and a translation table to correlate the input files with the abbreviations from the correlation plot.

Examples

```
data(SNPhood.o, package="SNPhood")
# Plot directly, using Pearson correlation
SNPhood.o = plotAndCalculateCorrelationDatasets(SNPhood.o)
# Plot to a PDF file
SNPhood.o = plotAndCalculateCorrelationDatasets(SNPhood.o, fileToPlot = "res.pdf")
# Using Spearman correlation instead of Pearson
SNPhood.o = plotAndCalculateCorrelationDatasets(SNPhood.o, corMeasure = "spearman")
```

---

plotAndCalculateWeakAndStrongGenotype

Visualizes and calculates strong and weak genotypes.

Description

The function `plotAndCalculateWeakAndStrongGenotype` finds the strongest and weakest genotypes based on reads extracted around each region. Strong and weak genotypes are found using the reads extracted from SNPhood and their corresponding genotypes as found by the function `associateGenotypes` Note the reads have to be merged using the function `mergeReadGroups` before running this function.

Usage

```
plotAndCalculateWeakAndStrongGenotype(SNPhood.o, normalize = TRUE,
  nClustersVec = 3, fileToPlot = NULL, verbose = FALSE)
```

Arguments

`SNPhood.o`	Object of class SNPhood
`normalize`	Logical(1). Default TRUE. Should a normalization be done on the counts/enrichments values before clustering? If set to TRUE, a normalization procedure based on subtracting the mean dividing by standard deviation for each region is performed. For more details, see the vignette.
`nClustersVec`	Numeric. Default 2. The number of clusters the data should be divided into. This can either be a vector or a single value. if multiple clusters are specified, multiple clustering analyses will be performed and for each of them, a plot is produced. make sure to specify the parameter fileToPlot in that case; otherwise, only the last plot may be visible.
`fileToPlot`	Character(1) or `NULL`. Default `NULL`. Filename of the PDF file for the output plots. If set to `NULL`, plots will be plotted to the currently active device.
`verbose`	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Modified SNPhood object with the results of the analysis stored in the object. Specifically, a matrix for average reads per SNP for datasets which have strong and weak genotypes, respectively, are stored in the slot additionalResults$genotype. The SNPs which have invariant genotypes across all the samples being analyzed are also saved. In addition, clustering on the strong and weak genotype read mateices are reportd as in the function plotAndClusterMatrix.

Examples

```
data(SNPhood.o, package="SNPhood")
SNPhood_merged.o = mergeReadGroups(SNPhood.o)
SNPhood_merged.o = plotAndCalculateWeakAndStrongGenotype(SNPhood_merged.o, nClustersVec = 6)
SNPhood_merged.o = plotAndCalculateWeakAndStrongGenotype(SNPhood_merged.o, nClustersVec = 2:6, verbose = FAL
```

---

plotAndClusterMatrix *Clustering of read counts or enrichmens across bins for a specific dataset and read group*

Description

plotAndClusterMatrix can be used to cluster regions such as SNPs based on their local neighbourhood. The underlying clustering is done using partitioning around medoids (PAM). For more details, see the vignette.

Usage

```
plotAndClusterMatrix(SNPhood.o, readGroup, dataset, nClustersVec = 3,
  normalize = TRUE, clustersToPlot = NULL, fileToPlot = NULL,
  verbose = FALSE, ...)
```

Arguments

SNPhood.o	Object of class SNPhood
readGroup	Character(1). Default NULL. Read group that should be plotted, specified by its name as obtained by the function annotationReadGroups). If only one read group is defined in the object, this may also be NULL for user conveniance.
dataset	Numeric(1) or Character(1). Single dataset that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or its annotation (name must appear in the dataset names as obtained via the function annotationDatasets).
nClustersVec	Numeric. Default 2. The number of clusters the data should be divided into. This can either be a vector or a single value. if multiple clusters are specified, multiple clustering analyses will be performed and for each of them, a plot is produced. make sure to specify the parameter fileToPlot in that case; otherwise, only the last plot may be visible.
normalize	Logical(1). Default TRUE. Should a normalization be done on the counts/enrichments values before clustering? If set to TRUE, a normalization procedure based on subtracting the mean dividing by standard deviation for each region is performed. For more details, see the vignette.

clustersToPlot	Integer. Default NULL. Vector of clusters that should be plotted. If set to NULL, all clusters from the clustering result will be plotted. Otherwise, only the clusters as specified by the user are plotted, omitting regions belonging to other clusters. This is useful to, for example, only display regions that show a bin-dependent pattern and are not invariant across the whole user region.
fileToPlot	Character(1) or NULL. Default NULL. Filename of the PDF file for the output plots. If set to NULL, plots will be plotted to the currently active device.
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?
...	Additional graphical parameters that can be used to modify the output of the function levelplot (panel.levelplot). See ?levelplot for details.

Value

The clustering reports the cluster in which each SNP falls, the average silhouette for pam clustering, plots for the clustered reads and a summary plot of average reads per cluster across the region being analyzed.

Examples

```
data(SNPhood.o, package="SNPhood")
SNPhood.o = plotAndClusterMatrix(SNPhood.o, readGroup = "paternal", dataset = 1, nClustersVec = c(3:6))
SNPhood.o = plotAndClusterMatrix(SNPhood.o, readGroup = "paternal", dataset = 1, normalize = FALSE)
```

---

plotAndSummarizeAllelicBiasTest

Summarize the allelic bias analysis across SNP regions and bins and visualize some of the results.

Description

plotAndSummarizeAllelicBiasTest summarizes the allelic bias test across SNP regions and bins by calculating various summary statistics. See the Vignette for more details. TODO

Usage

```
plotAndSummarizeAllelicBiasTest(SNPhood.o, signThreshold = 0.05,
  fileToPlot = NULL)
```

Arguments

SNPhood.o	Object of class SNPhood
signThreshold	Numeric(1). Default 0.05. The significance threshold (such as p-value or FDR threshold). Must be between 0 and 1. If the parameter belongs to a plotting function, a horizontal line is drawn at the chosen value. For the allelic bias summary plots, p-values below this threshold and the corresponding allelic fractions are highlighted.
fileToPlot	Character(1) or NULL. Default NULL. Filename of the PDF file for the output plots. If set to NULL, plots will be plotted to the currently active device.

Value

A named list with various elements, each of which summarizes the allelic bias tests with a different focus. TODO

Examples

```
data(SNPhood, package="SNPhood")
SNPhood.o = testForAllelicBiases (SNPhood.o, readGroups = c("maternal", "paternal"))
SNPhood.o = plotAndSummarizeAllelicBiasTest(SNPhood.o)
```

`plotBinCounts` *Visualize counts or enrichment for a particular region across bins, datasets, and read groups.*

Description

`plotBinCounts` visualizes counts or enrichment for a particular region across bins, datasets, and read groups. Many graphical parameters can be adjusted to suit the needs of the user, see below.

Usage

```
plotBinCounts(SNPhood.o, regions = 1, readGroups = NULL, datasets = NULL,
  readGroupColors = NULL, ylim = NULL, addGenotype = TRUE,
  plotGenotypeRatio = FALSE, addTitle = TRUE, colorPalette = "Set1",
  printPlot = TRUE, fileToPlot = NULL, maxWidthLabels = NULL,
  verbose = FALSE)
```

Arguments

`SNPhood.o`	Object of class `SNPhood`
`regions`	Numeric or Character or NULL. Default NULL. Regions that should be plotted, either specified as integer (such as 1, value must be between 1 and the total number of regions as defined in the object) or their annotation (name must appear in the region names as obtained via the function `annotationRegions`). If set to NULL, all regions will be considered.
`readGroups`	Character or NULL. Default NULL. Read groups that should be plotted, specified by their name as obtained by the function `annotationReadGroups`). If set to NULL, all read groups will be considered.
`datasets`	Numeric or Character or NULL. Default NULL. Datasets that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or their annotation (name must appear in the dataset names as obtained via the function `annotationDatasets`). If set to NULL, all datasets will be considered.
`readGroupColors`	Character or NULL. Default NULL. Colors of the read groups that appear in the plot. If set to NULL, colors will be set automatically. The length of the vector must equal the **total** number of read groups that are defined in the `SNPhood` object.
`ylim`	Numeric(2). Default NULL. Range of the y-axis, as specified by a minimum and a maximum value. See ?ylim for details.

addGenotype Logical(1). Default TRUE. Should the genotype distribution for each read group at the original user position be displayed in the legend in addition? See the Vignette for more details how this distribution is determined.

plotGenotypeRatio
Logical(1). Default FALSE. Should the ratio of the genotypes be plotted instead of the count or enrichment values? Only applicable if the number of read groups to be plotted is 2 and if one region is plotted. Setting this parameter to TRUE may result in ratios across bins that are interrupted due to zero counts (and a resulting division by zero, which can therefore not be displayed). Also, ratios cannot be plotted if the genotype for the selected regions could not be determined due to the lack of reads overlapping with the particular region (see the Vignette for details).

addTitle Logical(1). Default TRUE. Should the plot contain a title that summarizes the genomic region that is visualized?

colorPalette Character(1). Default "Set1". Name of the palette from the RColorBrewer package from the qualitative palettes for the colors of the datasets that are plotted. Allowed palette names are "Accent", "Dark2", "Paired", "Pastel1", "Pastel2", "Set1", "Set2", and "Set3". Colors for the datasets are then determined automatically from the given palette name (from left to right, depending on the number of datasets to be plotted). The colors for the read groups within each datasets are based on the colors for the dataset, but with different saturation values.

printPlot Logical(1). Default TRUE. Should the plots be printed? Only relevant if fileToPlot is set to NULL; otherwise, the plots are always printed to the output file.

fileToPlot Character(1) or NULL. Default NULL. Filename of the PDF file for the output plots. If set to NULL, plots will be plotted to the currently active device.

maxWidthLabels Numeric(1). Default NULL. Maximum width of the legend labels in number of characters. If the width of the legend labels are longer, they are shortened. Set to NULL to not shorten labels.

verbose Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

the generated **ggplot2** plot(s) as list for further processing. May contain multiple plots, depending on the function. The plot(s) can then be plotted individually or modified arbitrarily as the user wants. For example, if multiple plots are returned and the plots have been saved in a variable called plots.l, simply type plots.l[[1]] to view the first plot.

Examples

```
data(SNPhood.o, package="SNPhood")

# Plot the first region, all parameters with their default values
plot = plotBinCounts(SNPhood.o)

# Plot the second region for the first dataset, using specific colors for the read groups.
plot = plotBinCounts(SNPhood.o, regions = 2, dataset = 1, readGroupColors = c("red","blue","gray"))

# Plot the first region for the first dataset and the genotype ratio. Save the plot in a variable
plot = plotBinCounts(SNPhood.o, regions = 1, readGroups = c("maternal", "paternal"), dataset = 1, plotGenotyp

#' # Plot all regions for the first dataset and aggregate. Save the plot in a variable
```

```
plot = plotBinCounts(SNPhood.o, regions = NULL, readGroups = c("maternal", "paternal"), dataset = 1)
```

---

plotClusterAverage *Visualize average enrichment per cluster*

Description

plotClusterAverage visualizes the average reads per cluster. Note that the function plotAndClusterMatrix has to be executed before plotClusterAverage is called for the same read group and dataset

Usage

```
plotClusterAverage(SNPhood.o, readGroup, dataset, fileToPlot = NULL,
  returnOnlyPlotNotObject = FALSE, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
readGroup	Character(1). Default NULL. Read group that should be plotted, specified by its name as obtained by the function annotationReadGroups). If only one read group is defined in the object, this may also be NULL for user conveniance.
dataset	Numeric(1) or Character(1). Single dataset that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or its annotation (name must appear in the dataset names as obtained via the function annotationDatasets).
fileToPlot	Character(1) or NULL. Default NULL. Filename of the PDF file for the output plots. If set to NULL, plots will be plotted to the currently active device.
returnOnlyPlotNotObject	Logical(1). Default FALSE. If set to TRUE, only the plots are returned but not the actual object. Otherwise, for consistancy among the various visualization functions, the SNPhood object is always returned, while the plots are either written to a PDF file as specified by the parameter fileToPlot and/or to the currently active graphics device (i.e., the console usually)
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

the generated **ggplot2** plot(s) as list for further processing. May contain multiple plots, depending on the function. The plot(s) can then be plotted individually or modified arbitrarily as the user wants. For example, if multiple plots are returned and the plots have been saved in a variable called plots.l, simply type plots.l[[1]] to view the first plot.

See Also

plotAndClusterMatrix

Examples

```
data(SNPhood.o, package="SNPhood")
plot = plotClusterAverage(SNPhood.o, readGroup = "paternal", dataset = 1)
```

plotFDRResults *Graphically summarize the results of the allelic bias analysis for a specific dataset and region.*

Description

plotAllelicBiasResults graphically summarizes the results of the allelic bias analysis for a specific dataset and region.

Usage

```
plotFDRResults(SNPhood.o, dataset, FDRThreshold = NULL, fileToPlot = NULL,
  printPlot = TRUE, returnOnlyPlotNotObject = FALSE, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
dataset	Numeric(1) or Character(1). Single dataset that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or its annotation (name must appear in the dataset names as obtained via the function annotationDatasets).
FDRThreshold	Numeric(1) or NULL. Default NULL. If set to a value between 0 and 1, a horizontal line will be drawn in the FDR summary plot to indicate at which p-value threshold the FDR reaches the user-defined value. Additionally, the maximum p-value threshold from the FDR summary data will be printed for which thr FDR is below the specified threshold.
fileToPlot	Character(1) or NULL. Default NULL. Filename of the PDF file for the output plots. If set to NULL, plots will be plotted to the currently active device.
printPlot	Logical(1). Default TRUE. Should the plots be printed? Only relevant if fileToPlot is set to NULL; otherwise, the plots are always printed to the output file.
returnOnlyPlotNotObject	Logical(1). Default FALSE. If set to TRUE, only the plots are returned but not the actual object. Otherwise, for consistancy among the various visualization functions, the SNPhood object is always returned, while the plots are either written to a PDF file as specified by the parameter fileToPlot and/or to the currently active graphics device (i.e., the console usually)
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

the generated **ggplot2** plot(s) as list for further processing. May contain multiple plots, depending on the function. The plot(s) can then be plotted individually or modified arbitrarily as the user wants. For example, if multiple plots are returned and the plots have been saved in a variable called plots.l, simply type plots.l[[1]] to view the first plot.

Examples

```
data(SNPhood.o, package="SNPhood")
SNPhood.o = testForAllelicBiases(SNPhood.o, readGroups = c("maternal", "paternal"))
# Plot FDR results for first dataset
plotFDRResults(SNPhood.o, annotationDatasets(SNPhood.o)[1], FDRThreshold = NULL, fileToPlot = NULL)

# Plot FDR results for second dataset, save in file and also detemrine p-value treshold for which the FDR is be
plotFDRResults(SNPhood.o, annotationDatasets(SNPhood.o)[1], FDRThreshold = 0.1, fileToPlot = "FDR_summary.pd
```

---

plotGenotypesPerCluster

Visualize average counts/enrichment based on strong and weak genotypes.

Description

The function `plotGenotypesPerCluster` plots average clusters per genotype based on the clustering results of the strong an weak genotype analysis (see `plotAndCalculateWeakAndStrongGenotype`), which has to be executed before.

Usage

```
plotGenotypesPerCluster(SNPhood.o, printBinLabels = TRUE, fileToPlot = NULL,
  printPlot = TRUE, returnOnlyPlotNotObject = FALSE, verbose = FALSE)
```

Arguments

`SNPhood.o`	Object of class `SNPhood`
`printBinLabels`	Logical(1). Default TRUE. Should the bin labels be printed? If multiple clusters are plotted simultaenously, bin labels might overlap, in which case `printBinLabels` can be set to FALSE.
`fileToPlot`	Character(1) or `NULL`. Default `NULL`. Filename of the PDF file for the output plots. If set to `NULL`, plots will be plotted to the currently active device.
`printPlot`	Logical(1). Default TRUE. Should the plots be printed? Only relevant if `fileToPlot` is set to NULL; otherwise, the plots are always printed to the output file.
`returnOnlyPlotNotObject`	Logical(1). Default FALSE. If set to TRUE, only the plots are returned but not the actual object. Otherwise, for consistancy among the various visualization functions, the `SNPhood` object is always returned, while the plots are either written to a PDF file as specified by the parameter `fileToPlot` and/or to the currently active graphics device (i.e., the console usually)
`verbose`	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

the generated **ggplot2** plot(s) as list for further processing. May contain multiple plots, depending on the function. The plot(s) can then be plotted individually or modified arbitrarily as the user wants. For example, if multiple plots are returned and the plots have been saved in a variable called plots.l, simply type plots.l[[1]] to view the first plot.

See Also

plotAndCalculateWeakAndStrongGenotype

Examples

```
data(SNPhood.o, package="SNPhood")
SNPhood_merged.o = mergeReadGroups(SNPhood.o)
SNPhood_merged.o = plotAndCalculateWeakAndStrongGenotype(SNPhood_merged.o)
plot = plotGenotypesPerCluster(SNPhood_merged.o, printPlot = FALSE)
```

---

plotGenotypesPerSNP *Plot genotype frequencies of regions across datasets.*

Description

Creates bar plots for the distribution of genotype frequencies of regions across individuals.

Usage

```
plotGenotypesPerSNP(SNPhood.o, regions = NULL, fileToPlot = NULL,
  returnOnlyPlotNotObject = FALSE, verbose = FALSE)
```

Arguments

SNPhood.o	Object of class SNPhood
regions	Numeric or Character or NULL. Default NULL. Regions that should be plotted, either specified as integer (such as 1, value must be between 1 and the total number of regions as defined in the object) or their annotation (name must appear in the region names as obtained via the function annotationRegions). If set to NULL, all regions will be considered.
fileToPlot	Character(1) or NULL. Default NULL. Filename of the PDF file for the output plots. If set to NULL, plots will be plotted to the currently active device.
returnOnlyPlotNotObject	Logical(1). Default FALSE. If set to TRUE, only the plots are returned but not the actual object. Otherwise, for consistancy among the various visualization functions, the SNPhood object is always returned, while the plots are either written to a PDF file as specified by the parameter fileToPlot and/or to the currently active graphics device (i.e., the console usually)
verbose	Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

the generated **ggplot2** plot(s) as list for further processing. May contain multiple plots, depending on the function. The plot(s) can then be plotted individually or modified arbitrarily as the user wants. For example, if multiple plots are returned and the plots have been saved in a variable called plots.l, simply type plots.l[[1]] to view the first plot.

See Also

plotAndClusterMatrix

Examples

```
data(SNPhood.o, package="SNPhood")
plot = plotGenotypesPerSNP(SNPhood.o, regions=1:20)
```

`plotRegionCounts` *Visualize the raw read counts across regions or a user-defined genomic range*

Description

`plotBinCounts` visualizes the raw read counts (i.e., before binning user regions) across regions or a user-defined genomic range. Note that only the results of a particular chromosome can be visualized. It is therefore only possible if the regions to be visualized are located on one particular chromosome; otherwise, an error is thrown.

Usage

```
plotRegionCounts(SNPhood.o, regions = NULL, datasets = NULL,
  readGroups = NULL, mergeReadGroupCounts = FALSE, plotChr = NULL,
  plotStartPos = NULL, plotEndPos = NULL, ylim = NULL,
  plotRegionBoundaries = FALSE, plotRegionLabels = FALSE,
  maxWidthLabels = NULL, colorPalette = "Set1", sizePoints = 4,
  type = "p", printPlot = TRUE, fileToPlot = NULL, verbose = FALSE)
```

Arguments

`SNPhood.o`	Object of class `SNPhood`
`regions`	Numeric or Character or NULL. Default NULL. Regions that should be plotted, either specified as integer (such as 1, value must be between 1 and the total number of regions as defined in the object) or their annotation (name must appear in the region names as obtained via the function `annotationRegions`). If set to NULL, all regions will be considered.
`datasets`	Numeric or Character or NULL. Default NULL. Datasets that should be used for plotting, either specified as integer (such as 1, value must be between 1 and the total number of datasets as defined in the object) or their annotation (name must appear in the dataset names as obtained via the function `annotationDatasets`). If set to NULL, all datasets will be considered.
`readGroups`	Character or NULL. Default NULL. Read groups that should be plotted, specified by their name as obtained by the function `annotationReadGroups`). If set to NULL, all read groups will be considered.
`mergeReadGroupCounts`	Logical(1). Default FALSE. Should the read groups be merged for visualization purposes?
`plotChr`	Character(1) or NULL. Default NULL. The name of the chromosome for which the visualization should be done. Must be a valid chromosome name. If set to NULL, other parameters (such as`regions`) determine which genomic region should be plotted.

`plotStartPos` Character(1) or NULL. Default NULL. The start coordinates for which the visualization should be done. Must be a valid number with respect to the chromosome it refers to. If set to NULL and the parameter `plotChr` is not NULL, the start coordinates are set to 1.

`plotEndPos` Character(1) or NULL. Default NULL. The end coordinates for which the visualization should be done. Must be a valid number with respect to the chromosome it refers to. If set to NULL and the parameter `plotChr` is not NULL, the end coordinates are determined automatically and the full chromosome will be plotted.

`ylim` Numeric(2). Default NULL. Range of the y-axis, as specified by a minimum and a maximum value. See ?ylim for details.

`plotRegionBoundaries`
Logical(1). Default FALSE. Should the region boundaries be drawn in the plot? If set to TRUE, two vertical lines will be drawn for each region, corresponding to the region boundaries upstream and downstream of the SNP. This visual aid may help to judge the size of the regions and overlaps among regions.

`plotRegionLabels`
Logical(1). Should the annotation of the regions be drawn vertically below the x axis? If many regions are plotted, labels may overlap; however, for a few regions, this is usually not a problem.

`maxWidthLabels` Numeric(1). Default NULL. Maximum width of the legend labels in number of characters. If the width of the legend labels are longer, they are shortened. Set to NULL to not shorten labels.

`colorPalette` Character(1). Default "Set1". Name of the palette from the `RColorBrewer` package from the qualitative palettes for the colors of the datasets that are plotted. Allowed palette names are "Accent", "Dark2", "Paired", "Pastel1", "Pastel2", "Set1", "Set2", and "Set3". Colors for the datasets are then determined automatically from the given palette name (from left to right, depending on the number of datasets to be plotted). The colors for the read groups within each datasets are based on the colors for the dataset, but with different saturation values.

`sizePoints` Numeric(1). Default 4. Size of the points that are drawn in the plot (if type is set to the default value of "p"). This parameter has no effect if type is set to "l".

`type` Character(1). "p" or "l". Default "p". What type of plot should be drawn, points ("p") or lines ("l")?

`printPlot` Logical(1). Default TRUE. Should the plots be printed? Only relevant if `fileToPlot` is set to NULL; otherwise, the plots are always printed to the output file.

`fileToPlot` Character(1) or `NULL`. Default `NULL`. Filename of the PDF file for the output plots. If set to `NULL`, plots will be plotted to the currently active device.

`verbose` Logical(1). Default FALSE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

the generated **ggplot2** plot(s) as list for further processing. May contain multiple plots, depending on the function. The plot(s) can then be plotted individually or modified arbitrarily as the user wants. For example, if multiple plots are returned and the plots have been saved in a variable called plots.l, simply type plots.l[[1]] to view the first plot.

Examples

```
data(SNPhood.o, package="SNPhood")

# Plot the read counts for the first ten regions
plot = plotRegionCounts(SNPhood.o, regions = 1:10)

# Plot the read counts for the full chr21
plot = plotRegionCounts(SNPhood.o, plotChr = "chr21")

# Plot the read counts for the full chr21, merge read group counts and decrease the point size
plot = plotRegionCounts(SNPhood.o, plotChr = "chr21", sizePoints = 2, mergeReadGroupCounts = TRUE)
```

---

renameBins *Rename bins.*

Description

renameBins renames bins from a *SNPhood* object.

Usage

```
renameBins(SNPhood.o, newBinsMapping, verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
newBinsMapping	Named list. For clarity of mapping, the names of the list must be the currently defined bin names, and the values of each element the corresponding new ones.
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

an object of class SNPhood with the requested bins being renamed.

See Also

deleteDatasets, deleteRegions

Examples

```
data(SNPhood.o, package="SNPhood")
mapping = list("Bin1_NEW")
names(mapping) = annotationBins(SNPhood.o)[1]
SNPhood_mod.o = renameBins(SNPhood.o, mapping)
```

renameDatasets *Rename datasets.*

Description

renameDatasets renames datasets from a *SNPhood* object.

Usage

```
renameDatasets(SNPhood.o, newDatasetsMapping, verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
newDatasetsMapping	Named list. Named list. For clarity of mapping, the names of the list must be the currently defined dataset names, and the values of each element the corresponding new ones.
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

an object of class SNPhood with the requested datasets being renamed.

See Also

renameBins, renameReadGroups, renameRegions

Examples

```
data(SNPhood.o, package="SNPhood")
mapping = list("Individual1", "Individual2")
names(mapping) = annotationDatasets(SNPhood.o)
SNPhood_mod.o = renameDatasets(SNPhood.o, mapping)
```

renameReadGroups *Rename read groups.*

Description

renameReadGroups renames a set of read groups from a *SNPhood* object.

Usage

```
renameReadGroups(SNPhood.o, newReadGroupsMapping, verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
newReadGroupsMapping	Named list. Named list. For clarity of mapping, the names of the list must be the currently defined read group names, and the values of each element the corresponding new ones.
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

an object of class SNPhood with the requested read groups being renamed.

See Also

renameBins, renameDatasets, renameRegions

Examples

```
data(SNPhood.o, package="SNPhood")
mapping = list("a", "b", "c")
names(mapping) = annotationReadGroups(SNPhood.o)
SNPhood_mod.o = renameReadGroups (SNPhood.o, mapping)
```

---

renameRegions *Rename regions.*

Description

renameRegions renames regions from a *SNPhood* object.

Usage

```
renameRegions(SNPhood.o, newRegionsMapping, verbose = TRUE)
```

Arguments

SNPhood.o	Object of class SNPhood
newRegionsMapping	Named list. For clarity of mapping, the names of the list must be the currently defined region names, and the values of each element the corresponding new ones.
verbose	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

An object of class SNPhood with the requested regions being renamed

See Also

renameBins, renameDatasets, renameReadGroups

Examples

```
data(SNPhood.o, package="SNPhood")
mapping = as.list(paste0(annotationRegions(SNPhood.o),".newName"))
names(mapping) = annotationRegions(SNPhood.o)
SNPhood_mod.o = renameRegions(SNPhood.o, mapping)
```

---

results *Get results of various analyses performed with a* SNPhood *object.*

Description

Return the results of a particular analyis that is stored in the SNPhood object.

Usage

```
results(SNPhood.o, type, elements = NULL)
```

Arguments

SNPhood.o	Object of class SNPhood
type	Character(1). Name of analyses one wants to retrieve the results for. Currently supported are "allelicBias", "clustering", "genotype" and "samplesCorrelation".
elements	Character. Default NULL. Which elements of the resulting list structure should be returned? If set to NULL, all elements will be returned. Otherwise, if names are provided, only the requested subset elements will be returned. If type equals "allelicBias", valid values are "pValue", "confIntervalMin", "confIntervalMax", "fractionEstimate", "background", "FDR_results", and "parameters". If type equals "clustering", valid values are the defined read groups in the object. If type equals "genotype", valid values are "strongGenotypes", "weakGenotypes", and "invariantGenotypes". If type equals "samplesCorrelation", valid values are "corTable", and "transl".

Value

A list with the results of the requested analysis and elements within.

Examples

```
data(SNPhood.o, package="SNPhood")
head(results(SNPhood.o, type="allelicBias", elements = "parameters"))
head(results(SNPhood.o, type="allelicBias"))
```

SNPhood *SNPhood: Investigate, quantify and visualise the epigenomic neighbourhood of SNPs using NGS data*

Description

For more information and an introduction to the package, see the two vignettes.

Value

Summary analyses and visualizations for the selected genomic regions with respect to, for example, their read counts, genotype, and allelic origin

SNPhood functions

analyzeSNPhood annotation annotationBins annotationBins2 annotationDatasets annotationReadGroups annotationRegions associateGenotypes collectFiles convertToAllelicFractions counts deleteDatasets deleteReadGroups deleteRegions enrichment getDefaultParameterList mergeReadGroups nBins nDatasets nReadGroups nRegions parameters plotAllelicBiasResults plotAllelicBiasResultsOverview plotAndCalculateCorrelationDatasets plotAndCalculateWeakAndStrongG plotAndClusterMatrix plotBinCounts plotClusterAverage plotGenotypesPerCluster plotGenotypesPerSNP plotRegionCounts renameBins renameDatasets renameReadGroups renameRegions results testForAllelicBiases

Contact Information

We value all the feedback that we receive and will try to reply in a timely manner. Please report any bug that you encounter as well as any feature request that you may have to <SNPhood@gmail.com>.

SNPhood-class *A class to represent, investigate, quantify and visualise the epigenomic neighbourhood of SNPs using NGS data*

Description

The class SNPhood stores read count-derived information from NGS files for a set of genomic regions of interest as well as associated metadata. It may additionally contain results of various subsequent analyses and statistical tests. See the description below or the Vignette for more details.

Slots

annotation Named list. Contains various annotation and metadata such as:

- regions: An object of class GenomicRanges that contains the user regions, including annotation and the position of the original user-provided position before creating regions and bins.
- genotype: A list one or two elements, both of which contain genotype-related information, either directly from the sequencing reads or externally derived from a VCF file using the function associateGenotypes.
- readGroups: The names of the read groups that are currently defined.

- files: Contains a named list with additional information about each processed file, such as type(signal or input), files(a vector of one or multiple filenames), and composite(TRUE or FALSE, indicating if this is a composite file from multiple individual files)

Elements from this slot can be retrieved with the accessor function annotation.

config Named list. Named list with the parameters as specified in the parameter list and additionally the specific parameters the function analyzeSNPhood was called with (such as onlyPrepareForDatasetCorrelation and input). Elements from this slot can be retrieved with the accessor function parameters.

readCountsUnbinned Named list (nested). Contains vectors of raw reads counts for each user region (before binning). The names of the list are the read groups and the filenames of the annotated datasets. Elements from this slot can be retrieved with the accessor function counts using type = "unbinned".

readCountsBinned Named list (nested). Each element contains a matrix of raw reads counts per user region and bin (i.e., after binning). The names of the list are the read groups and the filenames of the annotated datasets. Contains the raw read counts if normalization among all datasets ha sbeen performed (parameter normAmongEachOther is set to FALSE) and normalized read counts otherwise.

If read counts are recorded allele-specifically (in the following snippet paternal, maternal and ambiguous) for each group, the structure therefore may look like this:

- paternal:
  - dataset ID 1: Matrix of read counts for each user region across bins
  - dataset ID 2: Matrix of read counts for each user region across bins
  - ...
- maternal: See read group paternal, identical structure
- ambiguous: See read group paternal, identical structure

enrichmentBinned Named list. See the description for the slot readCountsBinned, with the only difference that this slot contains the enrichment after normalizing with an input rather than the read counts. If input normalization is turned off, this slot is empty.

additionalResults Named list. Contains additional information from subsequent analyses such as allelic bias tests or results of the genotype analysis. Initially empty. Different functions write the results in this slot. Elements from this slot can be retrieved with the accessor function results.

Constructors

Currently, a SNPhood object can only be constructed by executing the main function of the package, analyzeSNPhood.

Accessors

In the following code snippets, SNPhood.o is a SNPhood object and readGroupCur and datasetCur a particular read group and dataset as defined in SNPhood.o, respectively.

Get general annotation of a SNPhood object

annotation(SNPhood.o): Get the annotation information, a nested list with multiple components (see names(annotation(SNPhood.o))).

Get more specific annotation such as number and annotation of regions, datasets, bins, and read groups

nRegions(SNPhood.o): Get the number of user regions.

`nDatasets(SNPhood.o)`: Get the number of datasets.

`nBins(SNPhood.o)`: Get the number of bins.

`nReadGroups(SNPhood.o)`: Get the number of read groups.

`annotationRegions(SNPhood.o)`: Get the annotation of user regions.

`annotationDatasets(SNPhood.o)`: Get the annotation of datasets.

`annotationBins(SNPhood.o)`: Get the annotation of bins.

`annotationReadGroups(SNPhood.o)`: Get the annotation of read groups.

Get the parameters that were used for the analysis

`parameters(SNPhood.o)`: Get the parameter information, a nested list with multiple components (see names(parameters(SNPhood.o))).

Get counts before binning

`counts(SNPhood.o, type = "unbinned", readGroup = readGroupCur, dataset = datasetCur)`: Get the counts for each user region before binning. See ?counts for more details.

If applicable, get counts after binning

`counts(SNPhood.o, type = "binned", readGroup = readGroupCur, dataset = datasetCur)`: Get the counts for each user region after binning. See ?counts for more details.

If applicable, get enrichment after binning

`enrichment(SNPhood.o, type = "binned", readGroup = readGroupCur, dataset = datasetCur)`: Get the enrichment for each user region after binning. See ?enrichment for more details.

In addition, see the workflow vignette (browseVignettes(\"SNPhood\") for a full workflow that uses all accessors.

`SNPhood.o` *SNPhood example data*

Description

This dataset is an example dataset that can be used for exploring the `SNPhood` package. For more information, see the workflow vignette of the `SNPhood` and `SNPhoodData` package, respectively.

Value

an example `SNPhood` object from the `SNPhoodData` package with read counts for 174 genomic regions across 2 datasets, three read groups and 100 bins

`testForAllelicBiases` *Perform an allelic bias tests for each user region and bin.*

Description

`testForAllelicBiases` performs tests for allelic biases for each binned user region using binomial tests. For the parameter `readGroups`, the name of exactly two read groups must be provided for which allelic ratio tests should be performed. See the Vignette for more details.

Usage

```
testForAllelicBiases(SNPhood.o, readGroups, confLevel = 0.95,
  nullHypothesisFraction = 0.5, calcBackgroundDistr = TRUE,
  nRepetitions = 100, pValuesToTestBackground = c(1e-04, 5e-04, 0.001,
  0.005, seq(0.01, 1, 0.01)), verbose = TRUE)
```

Arguments

`SNPhood.o`	Object of class SNPhood
`readGroups`	Character or NULL. Default NULL. Read groups that should be plotted, specified by their name as obtained by the function `annotationReadGroups`). If set to NULL, all read groups will be considered.
`confLevel`	Numeric(1). Default 0.95. The confidence level for estimating the confidence intervals. Must be between 0 and 1.
`nullHypothesisFraction`	Numeric(1). Default 0.5. The expected probability under the null hypothesis of not having any bias. Must be between 0 and 1.
`calcBackgroundDistr`	Logical(1). Default `TRUE`. Should the background distribution be calculated? Note that this can be usually very time-consuming.
`nRepetitions`	Integer(1). Default 10. Number of repetitions for calculating the background distribution. Only relevant if `calcBackgroundDistr` is set to `TRUE`
`pValuesToTestBackground`	Numeric. Default c(0.0001, 0.0005, 0.001, 0.005, seq(0.01,1,0.01)). Set of p-values for which corresponding FDR values will be computed
`verbose`	Logical(1). Default TRUE. Should the verbose mode (i.e., diagnostic messages during execution of the script) be enabled?

Value

Object of class SNPhood with all the data from the allelic bias test stored in the slot `additionalResults`, which can be easily retrieved via the accessor function `results`. See the help pages of the result function (?results) or the vignette for details.

Examples

```
data(SNPhood.o, package="SNPhood")
## Perform the test without calculating the background distribution
SNPhood.o = testForAllelicBiases (SNPhood.o, readGroups = c("paternal","maternal"))
str(results(SNPhood.o, type="allelicBias"), list.len = 8)
```

```
## Check the parameters
results(SNPhood.o, type="allelicBias", elements = "parameters")
```

Index