Package 'CNVrd2'

June 4, 2026

Type Package

Title CNVrd2: a read depth-based method to detect and genotype complex common copy number variants from next generation sequencing data.

Version 1.51.0

Date 2014-10-04

Author Hoang Tan Nguyen, Tony R Merriman and Mik Black

Depends R (>= 3.0.0), methods, VariantAnnotation, parallel, rjags, ggplot2, gridExtra

VignetteBuilder knitr

Suggests knitr

Maintainer Hoang Tan Nguyen <hoangtannguyenvn@gmail.com>

Description CNVrd2 uses next-generation sequencing data to measure human gene copy number for multiple samples, indentify SNPs tagging copy number variants and detect copy number polymorphic genomic regions.

License GPL-2

Imports DNAcopy, IRanges, Rsamtools

biocViews CopyNumberVariation, SNP, Sequencing, Software, Coverage, LinkageDisequilibrium, Clustering.

Collate AllClasses.R AllGenerics.R countReadInWindow.R segmentSamples.R segmentSamplesUsingPopInformation.R identifyPolymorphicRegion.R plotPolymorphicRegion.R emnormalCNV.R groupCNVs.R searchGroupCNVs.R groupBayesianCNVs.R plotCNVrd2.R calculateLDSNPandCNV.R

URL https://github.com/hoangtn/CNVrd2

git_url https://git.bioconductor.org/packages/CNVrd2

git_branch devel

git_last_commit 1e605d7

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-06-04

Contents

`CNVrd2-package` *CNVrd2*

Description

CNVrd2 uses next-generation sequencing data to measure human gene copy number for multiple samples and indentify SNPs/INDELs which are in linkage disequilibrium with the gene copy number variation.

Details

Package:	CNVrd2
Type:	Package
Version:	1.0
Date:	2013-03-26
License:	GPL-2
Depends:	methods

Author(s)

Maintainer: Hoang Tan Nguyen <hoangtannguyenvn@gmail.com>

`calculateLDSNPandCNV` *calculateLDSNPandCNV*

Description

Identifying SNPs/INDELs being in linkage disequilibrium with CNV.

Usage

```
calculateLDSNPandCNV(sampleCNV = NULL, vcfFile = NULL,
 matrixGenotype = NULL, cnvColumn = NULL, popColumn = NULL,
 population = NULL, chr = NULL, hg = "hg19",
 st = NULL, en = NULL, nChunkForVcf = 10,
 codeSNP = c("Two", "Three"), codeCNV = c("CN", "ThreeGroup"),
 typeTest = c("All", "Dup", "Del"), parallel = FALSE)
```

Arguments

`sampleCNV`	A data frame with no missing data; The first column is samples, the other columns are copy number, population names. This object can be obtained from the clustering step (allGroups).
`vcfFile`	Name of zipped vcf file including SNPs/INDELs.
`matrixGenotype`	A matrix of SNPs/INDELs coded: 0 for 0|0 or 0/0; 1 for 0/1, 1/0, 0|1, 1|0; 2 for 1/1, 1|1. Rows are SNPs/INDELs and columns are samples. If users use this argument then the argument vcfFile is not necessary.
`cnvColumn`	A number indicating the column of CNV.
`popColumn`	A number indicating the column of a population being calculated LD values.
`population`	A character() vector indicating names of populations.
`chr`	A character string indicating a chromosome of genes/SNPs/INDELs.
`hg`	A character string indicating the version of a reference genome (default: hg19).
`st`	A number indicating a starting coordinate to read the vcf file.
`en`	A number indicating an ending coordinate to read the vcf file.
`nChunkForVcf`	A number indicating how many chunks users would like to divide the vcf file to read into R. It depends on users' computers. We usually use 10 to 50 for this argument.
`codeSNP`	A character string indicating a way to code unphased/phased SNPs/INDELs into numeric values (Two: 0, 1 or Three: 0, 1, 2).
`codeCNV`	A character string partial matching to one of: *"All"* will test copy-number counts and SNPs/INDELs, *"ThreeGroup"* will divide samples into three groups: deletion, normality and duplication.
`typeTest`	A character string partial matching to one of: "All" will test all copy-number status, "Dup/Del" will test for only duplicated/deleted versus normal.
`parallel`	A logical value indicating whether multicores are used or NOT (default: FALSE).

Value

`r2Andpvalues`: A data frame (or a list) with each row for a SNP/INDEL: all information includes p-values adjusted by the method of Benjamini and Hocheberg (1995) and r2 values between the SNP/INDEL and copy-number status.

Note

st and *en* must not be outside the coordinates of the VCF file.

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

References

Benjamini, Y., and Hochberg, Y., 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal Statistical Society. Series B (Methodological), 57, 289-300.

See Also

`fcgr3bMXL`

Examples

```
##Load data: fcgr3bMXL in CNVrd2 package############
data(fcgr3bMXL)
##Name a vcf file (vcfFile)
vcfFile <- system.file(package="CNVrd2", "extdata",
                       "chr1.161600000.161611000.vcf.gz")
##Make a data fame named sampleCNV including samples, CNs, population names

sampleCNV <- data.frame(copynumberGroups$allGroups[, c(1,2) ],rep("MXL", 58))

rownames(sampleCNV) <- substr(sampleCNV[, 1], 1, 7)
sampleCNV[, 1] <- rownames(sampleCNV)
##The first column must be the sample names
tagSNPandINDELofMXL <- calculateLDSNPandCNV(sampleCNV = sampleCNV,
                                vcfFile = vcfFile, cnvColumn = 2,
                                population = "MXL", popColumn = 3,
                                nChunkForVcf = 5, chr = "1",
                                st = 161600000, en = 161611000,
                                codeSNP= "Three", codeCNV = "ThreeGroup")
tagSNPandINDELofMXL[1:3,]
```

ccl3l1data *Data of CCL3L1 gene (The 1000 Genomes Project)*

Description

This data set includes: segmentation results, population information and CCL3L1 CN.

Usage

```
data(ccl3l1data)
```

Format

This is a data frame including four columns: *Name* (names of samples), *Pop* (names of populations), *SS* (segmentation scores of samples at the gene) and *CN* (CCL3L1 CN obtained by using a Bayesian clustering approach with European-ancestry as prior information).

References

www.1000genomes.org

clusteringCNVs-class *Class* "clusteringCNVs"

Description

This class is used to cluster segmentation scores into copy-number groups.

Objects from the Class

Objects can be created by calls of the form `new("clusteringCNVs", ...)`.

Slots

x: Object of class "numeric".

k: Object of class "numeric" indicating a number of groups.

p: Object of class "numericOrNULL" indicating groups' proportions.

m: Object of class "numericOrNULL" indicating groups' means.

sigma: Object of class "numericOrNULL" indicating groups' standard deviations.

small: Object of class "numeric" indicating the value to stop the iteration process of the EM algorithm.

nMax: Object of class "numeric" indicating a maximum number of iterations.

EV: Object of class "logical" indicating whether all groups having equal variances or not (default).

eee: Object of class "numeric" indicating a pseudo value of 0.

nmaxInit: Object of class "numeric" indicating a number of iterations to obtain initial values.

nChangeVariance: Object of class "numeric" indicating a number of times to change from unequal variances to equal variances (*"this option is used to avoid the EM algorithm being broken down if there is one (or a few) sample in a group"*).

verbose: Object of class "logical" indicating whether printing out all loops.

groupDistance: Object of class "numericOrNULL" indicating the distance between groups.

Methods

emnormalCNV signature(Object = "clusteringCNVs"): run the EM algorithm.

groupCNVs signature(Object = "clusteringCNVs"): cluster segmentation scores into groups.

searchGroupCNVs signature(Object = "clusteringCNVs"): identify a number of groups.

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

Examples

```
showClass("clusteringCNVs")
```

CNVrd2-class *Class* "CNVrd2"

Description

A class of reading BAM files into R and grouping read-count windows into similar segments.

Objects from the Class

Objects can be created by calls of the form new("CNVrd2", ...).

Slots

windows: Object of class "numeric" indicating a window size.

chr: Object of class "character" indicating the chromosome of the region.

st: Object of class "numeric" indicating the starting coordinate of the region.

en: Object of class "numeric" indicating the ending coordinate of the region.

dirBamFile: Object of class "character" indicating a directory of BAM files.

dirCoordinate: Object of class "character" indicating a directory where all the positions of mapped reads will be written out to prepare for the segmentation process.

genes: Object of class "numeric" indicating gene coordinates.

geneNames: Object of class "character" indicating names of genes.

Methods

countReadInWindow signature(Object = "CNVrd2"): Count reads in windows.

plotCNVrd2 signature(Object = "CNVrd2"): Plot traces of samples.

segmentSamples signature(Object = "CNVrd2"): Cluster windows of read counts into regions having similar signal values.

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

Examples

```
showClass("CNVrd2")
```

---

countReadInWindow *Obtain read counts in constant windows.*

Description

Counting, transfering and standardizing read counts for all windows of samples. If correctGC = TRUE then all read-count windows will be corrected by the method of Yoon et al. (2009).

Usage

```
countReadInWindow(Object, ...)
```

Arguments

Object	An object of class CNVrd2.
...	Further aguments.

Value

readCountMatrix: a matrix of read counts for all samples (rows).

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

References

Yoon, S., Xuan, Z., Makarov, V., Ye, K., Sebat, J., 2009. Sensitive and accurate detection of copy number variants using read depth of coverage. Genome research 19 (9), 1586-1592.

Examples

```
## Not run:
data(fcgr3bMXL)
bamFiles <- dir("Bam", pattern = ".bam$")
objectCNVrd2 <- new("CNVrd2", windows = 1000, chr = "chr1",
                    st = 161100001, en = 162100000,
                    dirBamFile = "Bam",
                    genes = c(161592986, 161601753),
                    geneNames = "3B")

readCountMatrix <- countReadInWindow(Object = objectCNVrd2, correctGC = TRUE)
readCountMatrix[1:3, 1:3]

## End(Not run)
```

countReadInWindow-methods *Method* countReadInWindow

Description

Method to count reads in windows

Methods

`signature(Object = "CNVrd2")` Count, stransfer and standardize read count in windows of samples

See Also

countReadInWindow

Examples

```
##data(fcgr3bMXL)
##readCountMatrix <- countReadInWindow(Object = objectCNVrd2, correctGC = TRUE)
##readCountMatrix[1:3, 1:3]
```

emnormalCNV *Implement the EM algorithm*

Description

This function is used to obtain the maximization likelihood estimation of normal mixture model by using the EM algorithm (Demster et al., 1977).

Usage

```
emnormalCNV(Object, ...)
```

Arguments

Object	An object of class clusteringCNVs.
...	Optional arguments

Value

loglk	Value of the likelihood function.
p	Proportions of groups.
m	Means of groups.
sigma	Standard deviations of groups.
count	A number of iteration to obtain convergence stage.
bic	See searchGroupCNVs.
z	Data frame of proportions of data in mixture components.

Note

In the package, the distance between two initial means of the two nearest neighbor groups was set groupDistance = 0.25 as a default value to obtain initial values (using the kmeans function in R).

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

References

Dempster, A. P., Laird, N. M., Rubin, D. B., 1977. Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological), 1-38.

See Also

searchGroupCNVs, groupCNVs

Examples

```
data(fcgr3bMXL)

sS <- resultSegment$segmentationScores
#########Histogram#############################
###View segmentation scores#################
hist(sS[, 1], 100)
##############################################
##Number of components#######################
###Make an object of clusteringCNVs class######
objectCluster <- new("clusteringCNVs",
                     x = sS[, 1], k = 4, EV = TRUE)

set.seed(123)
copynumberGroups <- groupCNVs(Object = objectCluster)
```

---

emnormalCNV-methods *Method* emnormalCNV

Description

Implement the Expectation Maximization

Methods

```
signature(Object = "clusteringCNVs")
```

fcgr3bMXL *MXL population data (The 1000 Genomes Project)*

Description

This data set includes: segmentation results "resultSegment", information copy number "copynumberGroups" of *FCGR3B* gene.

Usage

```
data(fcgr3bMXL)
```

Format

All results analysed by the CNVrd2 package.

References

www.1000genomes.org

groupBayesianCNVs *groupBayesianCNVs*

Description

Cluster segmentation scores into different groups by using prior information from one population.

Usage

```
groupBayesianCNVs(xData, nGroups, lambda0, sd0, alpha0, distanceBetweenGroups, inits = NULL,
  precisionOfGroupMeans = 3000, sdOftau = NULL, n.adapt = 100,
  nUpdate = 1000, n.iter = 20000, thin = 5, n.chains = 1,
  heidel.diag = FALSE, leftLimit = NULL, rightLimit = NULL)
```

Arguments

xData	a numeric vector of observations (segmentation scores).
nGroups	an integer indicating a number of groups.
lambda0	Prior means of groups.
sd0	Prior standard deviations of groups.
alpha0	Prior parameters for mixing proportions.
distanceBetweenGroups	Prior value for the distance between groups.
inits	A list of initial values of parameters.
precisionOfGroupMeans	Prior parameter of group means (default = 3000).

sdOftau	Prior parameter of the standard deviations of group precisions.
n.adapt	the number of iterations for adaptation (rjags's parameter)
nUpdate	the number of iterations for burn-in process.
n.iter	the number of iterations for sampling (rjags's parameter)
thin	thinning interval for monitors (rjags's parameter)
n.chains	the number of parallel chains for the model (rjags's parameter, default = 1)
heidel.diag	If heidel.diag = TRUE then Heidelberger and Welch's convergence diagnostic is used.
leftLimit	Values which are less than this value will be allocated to the smallest group.
rightLimit	Values which are larger than this value will be allocated to the largest group.

Details

This function assumes that users already know the information of groups' means, standard deviations; the distances between groups.

Value

mcmcChains	A list of mcarray objects for means, standard deviations, proportions
m1	Means of groups
s1	Standard deviations of groups
p1	Proportions of groups
allGroups	A data.frame includes samples and their corresponding groups
hTest	Results of Heidelberger and Welch's convergence diagnostic

Note

#####

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

References

Martyn Plummer (2013). rjags: Bayesian graphical models using MCMC. R package version 3-10. http://CRAN.R-project.org/package=rjags.

Lunn, David J., et al. WinBUGS-a Bayesian modelling framework: concepts, structure, and extensibility. Statistics and Computing 10.4 (2000): 325-337.

See Also

groupCNVs

Examples

```
## Not run:
data(ccl3l1data)

xyEuro <- ccl3l1data[grep("CEU|TSI|IBS|GBR|FIN", ccl3l1data[, 2]), ]

names(yEuro) <- rownames(xyEuro)

##Clustering European segmentation scores into group: 5 groups were chosen

objectClusterEuroCCL3L1 <- new("clusteringCNVs", x = yEuro, k = 5)

europeanCCL3L1Groups <- groupCNVs(Object = objectClusterEuroCCL3L1)

##Obtain prior information
#Means
lambda0 <- as.numeric(europeanCCL3L1Groups$m)
#SD
sdEM <- as.numeric(europeanCCL3L1Groups$sigma)
#Proportions
pEM <- as.numeric(europeanCCL3L1Groups$p)


###Calculate the distances between groups
for (ii in 2:5){print(lambda0[ii] - lambda0[ii-1])}

###All segmentation scores
ccl3l1X <- ccl3l1data$SS
names(ccl3l1X) <- as.character(ccl3l1data$Name)
range(ccl3l1X)



##Set prior information:
#prior for the sd of the means of groups:
#5 was set for the third group = 2 CN
sd <- c(1, 1, 5, 1, 1)
ccl3l1X <- sort(ccl3l1X)
###Data
xData <- ccl3l1X
###Number of groups
nGroups <- 10
###prior for means of groups
lambda0 <- lambda0
###Prior for mixing proportions
alpha0 <-  c(3, 29, 44, 18, 7,  5, rep(2, nGroups -length(pEM) -1))
##Prior for the distances between groups
distanceBetweenGroups = 0.485

sdEM = sdEM


##Adjust standard deviation for the fifth group
sdEM[5] <- sdEM[4]

 set.seed(123)
```

```
groupCCL3L1allPops <- groupBayesianCNVs(xData = xData, nGroups = nGroups,
                                        lambda0 = lambda0,
                                        sd0 = sdEM, alpha0 = alpha0,
                                        distanceBetweenGroups = distanceBetweenGroups,
                                        sdOftau = sd,
                                       rightLimit = 4)



## End(Not run)
```

`groupCNVs` *Cluster segmentation scores into groups.*

Description

Use the EM algorithm (Dempster et al., 1977) to cluster segmentation scores into various groups.

Usage

```
groupCNVs(Object, ...)
```

Arguments

`Object`	An object of class clusteringCNVs.
`...`	Further arguments.

Details

Users can set limits of segmentation scores: values being smaller than the left limit will be assigned to the smallest group and values being larger than righ limit will be assigned to the largest group.

Value

`allGroups`	Samples and their corresponding groups
`means`	Means of groups.
`sigma`	Variances of groups.
`p`	Proportions of groups in all data set.
`loglk`	Value of loglikehood function.

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

References

Dempster, A. P., Laird, N. M., Rubin, D. B., 1977. Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological), 1-38.

See Also

emnormalCNV, searchGroupCNVs

Examples

```
data("fcgr3bMXL")
#resultSegment <- segmentSamples(Object = objectCNVrd2, stdCntMatrix = readCountMatrix)
objectCluster <- new("clusteringCNVs",
                     x = resultSegment$segmentationScores[, 1], k = 4, EV = TRUE)

#searchGroupCNVs(Object = objectCluster)
copynumberGroups <- groupCNVs(Object = objectCluster)
```

---

groupCNVs-methods *Method* groupCNVs

Description

Method groupCNVs

Methods

signature(Object = "clusteringCNVs")

---

identifyPolymorphicRegion *Identity polymorphic regions.*

Description

Using quantile values to identify polymorphic regions.

Usage

```
identifyPolymorphicRegion(Object, ...)
```

Arguments

Object	An object of class CNVrd2
...	polymorphicRegionObjectAn object obtained from the process of identifying poplymorphic regions. Optional arguments.

Value

`putativeBoundary`
: resultant boundaries based on quantile thresholds.

`subRegionMatrix`
: segmentation-score matrix of sub-regions derived from the segmentation process.

`subRegion`
: sub-regions derived from the segmentation process.

`mQuantile`
: matrix of quantile values.

`mSD`
: data frame including subregions and their standard deviations.

`Vst`
: data frame of Vst between populations if VstTest = TRUE.

`SSofPolymorphicRegions`
: segmentation scores of polymorphic regions.

Note

Users can choose various quantile values and adjust different thresholds to obtain polymorphic regions.

To visualize more clearly polymorphic regions user can use the method plotPolymorphicRegion with the option *typePlot="SD"*.

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

See Also

`plotPolymorphicRegion`

Examples

```
## Not run:

fcgr3PolymorphicRegion <- identifyPolymorphicRegion(Object = objectCNVrd2,
                                                    segmentObject = resultSegment,
                                          thresholdForPolymorphicRegions = c(0.75, 0.25),
                                                    plotLegend = FALSE)



## End(Not run)
```

---

identifyPolymorphicRegion-methods

Methods for Function `identifyPolymorphicRegion`

Description

Methods for function `identifyPolymorphicRegion`

Methods

```
signature(Object = "CNVrd2")
```

numericOrNULL-class *Class "numericOrNULL"*

Description

Auxiliary classes; may contain either a numeric vector or NULL [or a call / data.frame or NULL, respectively].

Objects from the Class

A virtual Class: No objects may be created from it.

Methods

No methods defined with class "numericOrNULL" in the signature.

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

plotCNVrd2 *Plot traces of samples.*

Description

Plot traces of samples.

Usage

```
plotCNVrd2(Object, ...)
```

Arguments

Object	An object of class CNVrd2.
...	Optional arguments.

Value

Plot

Note

Users can plot multiple samples simultaneously.

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

Examples

```
data(fcgr3bMXL)
##Obtain all information of CNVs
allGroups <- copynumberGroups$allGroups
###Obtain names of duplicate samples
duplicatedSamples <- rownames(allGroups[allGroups[, 2] > 2,])
###Plot the first duplicate samples
par(mfrow = c(3, 2))
for (ii in duplicatedSamples[1:6])
plotCNVrd2(Object = objectCNVrd2,
           segmentObject = resultSegment,
           sampleName = ii)
```

---

plotCNVrd2-methods *Method* plotCNVrd2

Description

Method plotCNVrd2

Methods

signature(Object = "CNVrd2")

---

plotPolymorphicRegion *Plot polymorphic regions.*

Description

Plot polymorphic regions based on coordinates set by users.

Usage

```
plotPolymorphicRegion(Object, ...)
```

Arguments

Object	An object of class CNVrd2
...	polymorphicRegionObjectAn object obtained from the process of identifying poplymorphic regions. Optional arguments.

Value

putativeBoundary

Putative boundaries of polymorphic regions based on quantile values.

Note

Users can choose various quantile values and adjust different thresholds to obtain polymorphic regions.

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

See Also

identifyPolymorphicRegion

Examples

```
## Not run:

plotPolymorphicRegion(Object = objectCNVrd2, polymorphicRegionObject = fcgr3PolymorphicRegion,
                      xlim = c(161300000, 161800000), drawThresholds = TRUE,
                      thresholdForPolymorphicRegions = c(0.75, 0.25))



##Change thresholds
plotPolymorphicRegion(Object = objectCNVrd2, polymorphicRegionObject = fcgr3PolymorphicRegion,
                      xlim = c(161300000, 161800000), drawThresholds = TRUE,
                      thresholdForPolymorphicRegions = c(0.9, 0.1))


##Plot standard deviation

plotPolymorphicRegion(Object = objectCNVrd2, polymorphicRegionObject = fcgr3PolymorphicRegion,
                      xlim = c(161300000, 161800000), typePlot = "SD",
                      thresholdForPolymorphicRegions = c(0.75, 0.25))







## End(Not run)
```

---

plotPolymorphicRegion-methods

Methods for Function plotPolymorphicRegion

Description

Methods for function plotPolymorphicRegion

Methods

signature(Object = "CNVrd2")

searchGroupCNVs *Choose a number of CN groups*

Description

Choose a number of CN groups by using Bayesian Information Criterion (Schwarz, 1978).

Usage

```
searchGroupCNVs(Object, ...)
```

Arguments

Object	An object of class *clusteringCNVs*.
...	Optional arguments.

Value

Groups	General information.
nComponents	A suitable number of components.

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

References

Schwarz, G. , 1978. Estimating the dimension of a model. The Annals of Statistics 6(2), 461-464.

searchGroupCNVs-methods
Method searchGroupCNVs

Description

Method searchGroupCNVs

Methods

```
signature(Object = "clusteringCNVs")
```

segmentSamples *Implement the segmentation process*

Description

Segment read-count windows into region having similar signal values by using the DNAcopy package (Venkatraman and Olshen, 2007) and refine this process to obtain segmentation scores at genes.

Usage

```
segmentSamples(Object, ...)
```

Arguments

Object	An object of class CNVrd2.
...	Optional arguments.

Value

segmentResults	All results of the segmentation process.
segmentationScores	Segmentation scores of the gene(s) being measured.
observedReadCountRatios	Observed read-count ratios of genes. This value is a matrix of observed read-count ratios at genes if (only inputBamFile = TRUE).
stdCntMatrix	Matrix of read counts (standardized).

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

References

Venkatraman, E., Olshen, A. B., 2007. A faster circular binary segmentation algorithm for the analysis of array chg data. Bioinformatics 23 (6), 657-663.

See Also

countReadInWindow, DNAcopy

Examples

```
data(fcgr3bMXL)
## Not run: resultSegment <- segmentSamples(Object = objectCNVrd2, stdCntMatrix = readCountMatrix)
```

segmentSamples-methods

Method segmentSamples

Description

Method segmentSamples

Methods

signature(Object = "CNVrd2")

segmentSamplesUsingPopInformation

Implement the segmentation process for multiple populations

Description

Segment read-count windows into region having similar signal values by using the DNAcopy package (Venkatraman and Olshen, 2007) and refine this process to obtain segmentation scores at genes. Then, the function adjusts segmentation scores for multiple populations using a linear regression model.

Usage

```
segmentSamplesUsingPopInformation(Object, ...)
```

Arguments

Object	An object of class CNVrd2.
...	Optional arguments.

Value

segmentationScores
: All adjusted results of the segmentation process.

segmentationScoresFromSinglePops
: Segmentation scores of single populations.

segmentResults
: Un-adjusted segmentation scores

observedReadCountRatios
: Observed read-count ratios of genes. This value is a matrix of observed read-count ratios at genes if (only inputBamFile = TRUE).

stdCntMatrix
: Matrix of read counts (standardized).

Author(s)

Hoang Tan Nguyen, Tony R Merriman and MA Black. <hoangtannguyenvn@gmail.com>

References

Venkatraman, E., Olshen, A. B., 2007. A faster circular binary segmentation algorithm for the analysis of array chg data. Bioinformatics 23 (6), 657-663.

See Also

countReadInWindow, DNAcopy

---

segmentSamplesUsingPopInformation-methods
Method segmentSamplesUsingPopInformation

Description

Method segmentSamplesUsingPopInformation

Methods

signature(Object = "CNVrd2")

---

vectorORfactor-class *Class* "vectorORfactor"

Description

Auxiliary class

Objects from the Class

A virtual Class: No objects may be created from it.

Methods

No methods defined with class "vectorORfactor" in the signature.

Author(s)

Hoang Tan Nguyen, Tony Merriman and Mik Black. <hoangtannguyenvn@gmail.com>

Index