cn.mops

March 24, 2012

`CNVDetectionResult-class`
Class "CNVDetectionResult"

Description

S4 class for storing results of a CNV detection method.

Slots

The following slots are defined for CNVDetectionResult objects:

normalizedData The normalized data.

localAssessments The data to which the segmentation algorithm is applied. These can be z-Scores, ratios, log-ratios or I/NI calls.

individualCall The CNV call that the method provides for a specific sample

iniCall The CNV call that the method provides a specific segment.

posteriorProbs The posterior probabilities for different copy numbers.

cnvs The detected CNVs.

cnvr The detected CNV regions.

segmentation The segmentation of the reference sequence (sample-wise).

integerCopyNumber The most probable integer copy number.

params The parameters with which the method was run.

Methods

cnvr `signature(object = "CNVDetectionResult")`: ...

cnvs `signature(object = "CNVDetectionResult")`: ...

individualCall `signature(object = "CNVDetectionResult")`: ...

iniCall `signature(object = "CNVDetectionResult")`: ...

integerCopyNumber `signature(object = "CNVDetectionResult")`: ...

localAssessments `signature(object = "CNVDetectionResult")`: ...

normalizedData `signature(object = "CNVDetectionResult")`: ...

params `signature(object = "CNVDetectionResult")`: ...

plot `signature(x = "CNVDetectionResult", y = "missing"): ...`

posteriorProbs `signature(object = "CNVDetectionResult"): ...`

segmentation `signature(object = "CNVDetectionResult"): ...`

segplot `signature(object = "CNVDetectionResult"): ...`

show `signature(object = "CNVDetectionResult"): ...`

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
showClass("CNVDetectionResult")
```

`CNVRanges` *Genomic locations and indices of the simulated CNVs.*

Description

This data set gives the starts, ends, and the integer copy number of the simulated CNVs in the data set `XRanges` object.

Usage

```
CNVRanges
```

Format

A GRanges object with 20 rows and 40 value columns across 1 space.

Source

http://www.bioinf.jku.at/cnmops/cnmops.html.

References

Klambauer et al., cn.mops, *Unpublished method.*, 2011

X *A simulated data set for CNV detection from NGS data.*

Description

This data set gives the read counts of 40 samples in 5000 genomic locations. The rows correspond to genomic segments of 25kbp length and the columns to samples. An entry is the number of reads that map to the specific segment of the sample. The rownames contain the information of the genomic location - they are in the format refseqname_startposition_endposition. The simulated data contains CNVs given in the CNVRanges object. It was generated using distributions of read counts as they appear in real sequencing experiments. CNVs were implanted under the assumption that the expected read count is linear dependent on the copy number (e.g. in a certain genomic we expect

$$\lambda$$

reads for copy number 2, then we expect

$$2 \cdot \lambda$$

reads for copy number 4).

Usage

```
X
```

Format

A data matrix of 5000 rows and 40 columns.

Source

http://www.bioinf.jku.at/software/cnmops/cnmops.html.

References

Klambauer et al., cn.mops, *Unpublished method.*, 2011

XRanges *A simulated data set for CNV detection from NGS data.*

Description

This data set gives the read counts of 40 samples in 5000 genomic locations. The rows correspond to genomic segments of 25kbp length and the columns to samples. An entry is the number of reads that map to the specific segment of the sample. The "GRanges" object contains the name of the reference sequence, start and end position of the genomic segments. The simulated data contains CNVs given in the CNVRanges object. It was generated using distributions of read counts as they appear in real sequencing experiments. CNVs were implanted under the assumption that the expected read count is linear dependent on the copy number (e.g. in a certain genomic we expect

$$\lambda$$

reads for copy number 2, then we expect

$$2 \cdot \lambda$$

reads for copy number 4).

Usage

```
XRanges
```

Format

A GRanges object with 5000 rows and 40 value columns across 1 space.

Source

http://www.bioinf.jku.at/software/cnmops/cnmops.html.

References

Klambauer et al., cn.mops, *Unpublished method.*, 2011

---

cn.mops — *Performs the cn.mops algorithm for copy number detection in NGS data.*

Description

Performs the cn.mops algorithm for copy number detection in NGS data.

Usage

```
cn.mops(input,
  I = c(0.025, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4),
  classes = c("CN0", "CN1", "CN2", "CN3", "CN4", "CN5", "CN6", "CN7", "CN8"),
  priorImpact = 1, cyc = 20, parallel = 0,
  normType = "poisson", normQu = 0.25, norm = TRUE,
  upperThreshold = 0.5, lowerThreshold = -0.9,
  minWidth = 3, segAlgorithm = "fast", ...)
```

Arguments

input	Either an instance of "GRanges" or a raw data matrix, where columns are interpreted as samples and rows as genomic regions. An entry is the read count of a sample in the genomic region.
I	Vector positive real values that contain the expected fold change of the copy number classes. Length of this vector must be equal to the length of the "classes" parameter vector. For human copy number polymorphisms we suggest to use the default I = c(0.05,0.5,1,1.5,2,2.5,3,3.5,4).
classes	Vector of characters of the same length as the parameter vector "I". One vector element must be named "CN2". The names reflect the labels of the copy number classes. Default = c("CN0","CN1","CN2","CN3","CN4","CN5","CN6","CN7","CN8").
priorImpact	Positive real value that reflects how strong the prior assumption affects the result. The higher the value the more samples will be assumed to have copy number 2. Default = 1.
cyc	Positive integer that sets the number of cycles for the algorithm. Usually after less than 15 cycles convergence is reached. Default = 20.

parallel	How many cores are used for the computation. If set to zero than no parallelization is applied. The package "snow" has to be installed for this option. Default = 0.
normType	Mode of the normalization technique. Possible values are "mean","min","median","quant", "poisson" and "mode". Read counts will be scaled sample-wise. Default = "poisson".
normQu	Real value between 0 and 1. If the "normType" parameter is set to "quant" then this parameter sets the quantile that is used for the normalization. Default = 0.25.
norm	Logical that indicates whether normalization should be applied or not. Default = TRUE.
upperThreshold	Positive real value that sets the cut-off for copy number gains. All CNV calling values above this value will be called as "gain". The value should be set close to the log2 of the expected foldchange for copy number 3 or 4. Default = 0.5.
lowerThreshold	Negative real value that sets the cut-off for copy number losses. All CNV calling values below this value will be called as "loss". The value should be set close to the log2 of the expected foldchange for copy number 1 or 0. Default = -0.9.
minWidth	Positive integer that is exactly the parameter "min.width" of the "segment" function of "DNAcopy". minWidth is the minimum number of segments a CNV should span. Default = 4.
segAlgorithm	Which segmentation algorithm should be used. If set to "DNAcopy" circular binary segmentation is performed. Any other value will initiate the use of our fast segmentation algorithm. Default = "fast".
...	Additional parameters will be passed to the "DNAcopy" or the standard segmentation algorithm.

Value

An instance of "CNVDetectionResult".

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
cn.mops(XRanges)
```

---

cnvr *These generic function returns CNV regions of a CNV detection method stored in an instance of 'CNVDetectionResult-class'.*

Description

These generic function returns CNV regions of a CNV detection method stored in an instance of CNVDetectionResult-class.

Arguments

object	An instance of "CNVDetectionResult"

Value

cnvr returns a eturns a "GRanges" object containing the CNV regions.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
cnvr(r)
```

---

cnvs *These generic function returns CNVs of a CNV detection method stored in an instance of 'CNVDetectionResult-class'.*

Description

These generic function returns CNVs of a CNV detection method stored in an instance of CNVDetectionResult-class.

Arguments

object	An instance of "CNVDetectionResult"

Value

cnvs returns a eturns a "GRanges" object containing the CNVs.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
cnvs(r)
```

getReadCountsFromBAM

Generates the read counts from BAM Files. These counts are necessary for CNV detection methods based on depth of coverage information. Note that the function is much faster, if the BAM files have an index file. The index file is assumed to be in the same folder and have an identical file name except that ".bai" is appended.

Description

Generates the read counts from BAM Files. These counts are necessary for CNV detection methods based on depth of coverage information. Note that the function is much faster, if the BAM files have an index file. The index file is assumed to be in the same folder and have an identical file name except that ".bai" is appended.

Usage

```
getReadCountsFromBAM(BAMFiles, sampleNames, refSeqName,
  WL, mode = "unpaired")
```

Arguments

BAMFiles	BAMFiles
sampleNames	The corresponding sample names to the BAM Files.
refSeqName	Name of the reference sequence that should be analyzed. The name must appear in the header of the BAM file. If it is not given the function will select the first reference sequence that appears in the header of the BAM files.
WL	Windowlength. Length of the initial segmentation of the genome in basepairs. Should be chosen such that on the average 100 reads are contained in each segment. If not given, cn.mops will try to find an appropiate window length.
mode	Possible values are "paired" and "unpaired", whether the mapping algorithm was using a "paired" or "unpaired" strategy. Default = "unpaired".

Value

An instance of "GRanges", that contains the breakpoints of the initial segments and the raw read counts that were extracted from the BAM files. This object can be used as input for cn.mops and other CNV detection methods.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
BAMFiles <- list.files(system.file("extdata", package="cn.mops"),pattern=".bam$",
full.names=TRUE)
bamDataRanges <- getReadCountsFromBAM(BAMFiles,
sampleNames=paste("Sample",1:3),WL=5000)
```

getSegmentReadCountsFromBAM

Generates the read counts from BAM Files for predefined segments. This is the appropiate choice for exome sequencing data, where the bait regions, target regions or exons are the predefined segments. These counts are necessary for CNV detection methods based on depth of coverage information. Note that the function is much faster, if the BAM files have an index file. The index file is assumed to be in the same folder and have an identical file name except that ".bai" is appended.

Description

Generates the read counts from BAM Files for predefined segments. This is the appropiate choice for exome sequencing data, where the bait regions, target regions or exons are the predefined segments. These counts are necessary for CNV detection methods based on depth of coverage information. Note that the function is much faster, if the BAM files have an index file. The index file is assumed to be in the same folder and have an identical file name except that ".bai" is appended.

Usage

```
getSegmentReadCountsFromBAM(BAMFiles, GR, sampleNames,
  mode = "unpaired")
```

Arguments

BAMFiles	BAMFiles
sampleNames	The corresponding sample names to the BAM Files.
GR	A genomic ranges object that contains the genomic coordinates of the segments.
mode	Possible values are "paired" and "unpaired", whether the mapping algorithm was using a "paired" or "unpaired" strategy. Default = "unpaired".

Value

An instance of "GRanges", that contains the breakpoints of the initial segments and the raw read counts that were extracted from the BAM files. This object can be used as input for cn.mops and other CNV detection methods.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
BAMFiles <- list.files(system.file("extdata", package="cn.mops"),pattern=".bam$",
full.names=TRUE)
gr <- GRanges(c("20","20"),IRanges(c(60000,70000),c(70000,80000)))
bamDataRanges <- getSegmentReadCountsFromBAM(BAMFiles,GR=gr)
```

`individualCall` *These generic function returns the individual calls of a CNV detection method stored in an instance of 'CNVDetectionResult-class'.*

Description

These generic function returns the individual calls of a CNV detection method stored in an instance of `CNVDetectionResult-class`.

Arguments

`object`	An instance of "CNVDetectionResult"

Value

`individualCalls` returns a "GRanges" object containing the individual calls.

Author(s)

Guenter Klambauer `<klambauer@bioinf.jku.at>`

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
individualCall(r)
```

`iniCall` *These generic function returns the informative/non-informative call of a CNV detection method stored in an instance of 'CNVDetectionResult-class'. The I/NI call is a measure for a genomic segment across all samples, whether this segment is a CNV region (informative) or a normal genomic region (non-informative).*

Description

These generic function returns the informative/non-informative call of a CNV detection method stored in an instance of `CNVDetectionResult-class`. The I/NI call is a measure for a genomic segment across all samples, whether this segment is a CNV region (informative) or a normal genomic region (non-informative).

Arguments

`object`	An instance of "CNVDetectionResult"

Value

`iniCall` returns a "GRanges" object containing the individual calls.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
iniCall(r)
```

`integerCopyNumber` *These generic function returns the integer copy numbers of a CNV detection method stored in an instance of 'CNVDetectionResult-class'.*

Description

These generic function returns the integer copy numbers of a CNV detection method stored in an instance of `CNVDetectionResult-class`.

Arguments

`object`	An instance of "CNVDetectionResult"

Value

`integerCopyNumber` returns a eturns a "GRanges" object containing the integer copy numbers.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
integerCopyNumber(r)
```

`localAssessments` *These generic function returns the local assessments, i.e. signed individual informative/non-informative calls, of a CNV detection method stored in an instance of 'CNVDetectionResult-class'. For other CNV detection methods this can be (log-) ratios or z-scores.*

Description

These generic function returns the local assessments, i.e. signed individual informative/non-informative calls, of a CNV detection method stored in an instance of `CNVDetectionResult-class`. For other CNV detection methods this can be (log-) ratios or z-scores.

Arguments

`object`	An instance of "CNVDetectionResult"

Value

`localAssessments` returns a "GRanges" object containing the local assessments.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
localAssessments(r)
```

---

normalizeChromosomes

Normalize quantitative NGS data in order to make counts comparable over samples. Scales each samples' reads such that the coverage is even for all samples after normalization.

Description

Normalize quantitative NGS data in order to make counts comparable over samples. Scales each samples' reads such that the coverage is even for all samples after normalization.

Usage

```
normalizeChromosomes(X, chr, normType = "poisson",
  qu = 0.25, ploidy)
```

Arguments

`X`	Matrix of positive real values, where columns are interpreted as samples and rows as genomic regions. An entry is the read count of a sample in the genomic region.
`chr`	Character vector that has as many elements as "X" has rows. The vector assigns each genomic segment to a reference sequence (chromosome).
`normType`	Type of the normalization technique. Each samples' read counts are scaled such that the total number of reads is equal after normalization. By this parameter one can decide to which coverage (i.e. total reads) the read counts should be normalized. Possible choices are the minimal coverage ("min"), the mean or median coverage ("mean", "median") or any quantile ("quant"). If this parameter is set to the value "mode", the read counts are scaled such that each samples' most frequent value (the "mode") is equal after normalization. If the parameter is set to "poisson" the values are scaled such that the distribution is (rowwise) close to a Poisson distribution. Possible values are "mean","min","median","quant","poisson, and "mode". Default = "poisson".
`qu`	Real value between 0 and 1. Default = 0.25.
`ploidy`	An integer value for each sample or each column in the read count matrix. At least two samples must have a ploidy of 2. Default = "missing".

Value

A data matrix of normalized read counts with the same dimensions as the input matrix X.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
X.norm <- normalizeChromosomes(X)
```

`normalizeGenome` *Normalize quantitative NGS data in order to make counts comparable over samples. Scales each samples' reads such that the coverage is even for all samples after normalization.*

Description

Normalize quantitative NGS data in order to make counts comparable over samples. Scales each samples' reads such that the coverage is even for all samples after normalization.

Usage

```
normalizeGenome(X, normType = "poisson", qu = 0.25,
  ploidy)
```

Arguments

`X`	Matrix of positive real values, where columns are interpreted as samples and rows as genomic regions. An entry is the read count of a sample in the genomic region.
`normType`	normType Type of the normalization technique. Each samples' read counts are scaled such that the total number of reads is equal after normlization. By this parameter one can decide to which coverage (i.e. total reads) the read counts should be normalized. Possible choices are the minimal coverage ("min"), the mean or median coverage ("mean", "median") or any quantile ("quant"). If this parameter is set to the value "mode", the read counts are scaled such that each samples' most frequent value (the "mode") is equal after normalization. If the parameter is set to "poisson" the values are scaled such that the distribution is (rowwise) close to a Poisson distribution. Possible values are "mean","min","median","quant","poisso and "mode". Default = "poisson".
`qu`	Real value between 0 and 1. Default = 0.25.
`ploidy`	An integer value for each sample or each column in the read count matrix. At least two samples must have a ploidy of 2. Default = "missing".

Value

A data matrix of normalized read counts with the same dimensions as the input matrix X.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
X.norm <- normalizeGenome(X)
```

---

normalizedData — *These generic function returns the normalized data of a CNV detection method stored in an instance of 'CNVDetectionResult-class'.*

Description

These generic function returns the normalized data of a CNV detection method stored in an instance of CNVDetectionResult-class.

Arguments

object	An instance of "CNVDetectionResult".

Value

normalizedData returns a "GRanges" object containing the normalized data.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
normalizedData(r)
```

---

params — *These generic function returns the parameters of a CNV detection method stored in an instance of 'CNVDetectionResult-class'.*

Description

These generic function returns the parameters of a CNV detection method stored in an instance of CNVDetectionResult-class.

Arguments

object	An instance of "CNVDetectionResult"

Value

params returns a eturns a "GRanges" object containing the parameters.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
params(r)
```

---

plot *Plots a CNVDetectionResult*

Description

Plots read counts, call values and CNV calls in an identified CNV region.

Usage

```
## S4 method for signature 'CNVDetectionResult,missing'
plot(x,
             which,margin=c(10,10),toFile=FALSE)
```

Arguments

x	An instance of "CNVDetectionResult"
which	The index of the CNV region to be plotted.
margin	Vector of two positive integers that states how many segments left and right of the CNV region should be included in the plot. Default = c(10,10).
toFile	Logical value whether the output should be plotted to a file. Default = FALSE.

Value

Generates a CNV calling plot.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

`posteriorProbs` *These generic function returns the posterior probabilities of a CNV detection method stored in an instance of 'CNVDetectionResult-class'. The posterior probabilities are represented as a three dimensional array, where the three dimensions are segment, copy number and individual.*

Description

These generic function returns the posterior probabilities of a CNV detection method stored in an instance of `CNVDetectionResult-class`. The posterior probabilities are represented as a three dimensional array, where the three dimensions are segment, copy number and individual.

Arguments

`object`	An instance of "CNVDetectionResult"

Value

`posteriorProbs` returns a three dimensional array.

Author(s)

Guenter Klambauer `<klambauer@bioinf.jku.at>`

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
posteriorProbs(r)
```

`segment` *Performs a fast segmentation algorithm based on the cyber t test and the t statistics.*

Description

Performs a fast segmentation algorithm based on the cyber t test and the t statistics.

Usage

```
segment(x, alpha = 0.05, segMedianT = 0, minSeg = 3,
  eps = 0, delta = 20, maxInt = 40, squashing = 0,
  cyberWeight = 50, segPlot = TRUE, ...)
```

Arguments

x	Values to be segmented.
alpha	Real value between 0 and 1 is interpreted as the percentage of total points that are considered as initial breakpoints. An integer greater than 1 is interpreted as number of initial breakpoints. Default = 0.05.
segMedianT	Vector of length 2. Thresholds on the segment's median. Segments' medians above the first element are considered as gains and below the second value as losses.
minSeg	Minimum length of segments. Default = 3.
eps	Real value greater or equal zero. A breakpoint is only possible between to consecutive values of x that have a distance of at least "eps". Default = 0.
delta	Positive integer. A parameter to make the segmentation more efficient. If the statistics of a breakpoint lowers while extending the window, the algorithm extends the windows by "delta" more points until it stops. Default = 20.
maxInt	The maximum length of a segment left of the breakpoint and right of the breakpoint that is considered. Default = 40.
squashing	An experimental parameter that squashes the values "x" before segmentation. Should be left to zero, which means that squashing is not performed. Default = 0.
cyberWeight	The "nu" parameter of the cyber t-test. Default = 50.
segPlot	Logical indicating whether the result of the segmentation a algorithm should be plotted. Default = TRUE.
...	additional parameters passed to the plotting function.

Value

A data frame containing the segments.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
x <- rnorm(n=500,sd=0.5)
x[150:200] <- rnorm(n=51,mean=3,sd=0.5)
segment(x)
```

---

segmentation *These generic function returns segmentation of a CNV detection method stored in an instance of 'CNVDetectionResult-class'.*

Description

These generic function returns segmentation of a CNV detection method stored in an instance of CNVDetectionResult-class.

Arguments

object	An instance of "CNVDetectionResult"

Value

segmentation returns a eturns a "GRanges" object containing the segmentation.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
r <- cn.mops(X[1:100,1:5])
segmentation(r)
```

---

segplot *Plots the log normalized read counts and the detected segments for one sample on one reference sequence.*

Description

Plots the log normalized read counts and the detected segments for one sample on one reference sequence.

Arguments

r	An instance of "CNVDetectionResult"
seqname	The name of the reference sequence or chromosome to be plotted.
sampleIdx	The index of the sample as it appears in the read count matrix.

Value

Generates a segmentation plot.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Examples

```
data(cn.mops)
r <- cn.mops(X[1:200, ])
segplot(r,sampleIdx=1)
```

show *Displays the result object of a copy number detection method.*

Description

Displays method for S4 class CNVDetectionResult

Usage

```
## S4 method for signature 'CNVDetectionResult'
show(object)
```

Arguments

object	An instance of a "CNVDetectionResult".

Value

Displays the result object of a CNV detection method.

Author(s)

Guenter Klambauer <klambauer@bioinf.jku.at>

Index