Package 'PSCBS'

June 28, 2017

Version 0.63.0

Depends R (>= 3.2.0), utils

Imports R.methodsS3 (>= 1.7.1), R.oo (>= 1.21.0), R.utils (>= 2.5.0), R.cache (>= 0.12.0), matrixStats (>= 0.52.2), aroma.light (>= 2.4.0), DNAcopy (>= 1.42.0), listenv (>= 0.6.0), future (>= 1.5.0), parallel

Suggests Hmisc (>= 3.16-0), R.rsp (>= 0.41.0), R.devices (>= 2.15.1), ggplot2 (>= 2.2.1)

SuggestsNote BioC (>= 3.1), Recommended: Hmisc

VignetteBuilder R.rsp

Date 2017-06-27

Title Analysis of Parent-Specific DNA Copy Numbers

Description Segmentation of allele-specific DNA copy number data and detection of regions with abnormal copy number within each parental chromosome. Both tumor-normal paired and tumor-only analyses are supported.

License GPL (>= 2)

LazyLoad TRUE

ByteCompile TRUE

biocViews aCGH, CopyNumberVariants, SNP, Microarray, OneChannel, TwoChannel, Genetics

URL https://github.com/HenrikBengtsson/PSCBS

BugReports https://github.com/HenrikBengtsson/PSCBS/issues

RoxygenNote 6.0.1

NeedsCompilation no

Author Henrik Bengtsson [aut, cre, cph],
Pierre Neuvial [aut],
Venkatraman E. Seshan [aut],
Adam B. Olshen [aut],
Paul T. Spellman [aut],
Richard A. Olshen [aut]

Maintainer Henrik Bengtsson <henrikb@braju.com>

Repository CRAN

Date/Publication 2017-06-28 06:05:57 UTC

R topics documented:

PSCBS-package . . . 2
callSegmentationOutliers . . . 3
CBS . . . 4
findLargeGaps . . . 6
gapsToSegments.data.frame . . . 7
NonPairedPSCBS . . . 8
PairedPSCBS . . . 9
PSCBS . . . 11
segmentByCBS . . . 12
segmentByNonPairedPSCBS . . . 16
segmentByPairedPSCBS . . . 19

Index **25**

`PSCBS-package` *Package PSCBS*

Description

Segmentation of allele-specific DNA copy number data and detection of regions with abnormal copy number within each parental chromosome. Both tumor-normal paired and tumor-only analyses are supported..

This package should be considered to be in an alpha or beta phase. You should expect the API to be changing over time.

Installation and updates

To install this package, use `install.packages("PSCBS")`.

To get started

To get started, see:

1. Vignettes 'Parent-specific copy-number segmentation using Paired PSCBS' and 'Total copy-number segmentation using CBS'.
2. `segmentByCBS`() - segments total copy-numbers, or any other unimodal genomic signals, using the CBS method [3,4].
3. `segmentByPairedPSCBS`() - segments allele-specific tumor signal from a tumor *with* a matched normal using the Paired PSCBS method [1,2].
4. `segmentByNonPairedPSCBS`() - segments allele-specific tumor signal from a tumor *without* a matched normal using the Non-Paired PSCBS method adopted from [1,2].

How to cite

Please use [1] and [2] to cite when using Paired PSCBS, and [3] and [4] when using CBS. When using Non-Paired PSCBS, please cite [1] and [2] as well.

License

GPL (>= 2).

Author(s)

Henrik Bengtsson

References

[1] A.B. Olshen, H. Bengtsson, P. Neuvial, P.T. Spellman, R.A. Olshen, V.E. Seshan, *Parent-specific copy number in paired tumor-normal studies using circular binary segmentation*, Bioinformatics, 2011
[2] H. Bengtsson, P. Neuvial and T.P. Speed, *TumorBoost: Normalization of allele-specific tumor copy numbers from a single pair of tumor-normal genotyping microarrays*, BMC Bioinformatics, 2010
[3] A.B. Olshen, E.S. Venkatraman (aka Venkatraman E. Seshan), R. Lucito and M. Wigler, *Circular binary segmentation for the analysis of array-based DNA copy number data*, Biostatistics, 2004
[4] E.S. Venkatraman and A.B. Olshen, *A faster circular binary segmentation algorithm for the analysis of array CGH data*, Bioinformatics, 2007

---

callSegmentationOutliers

Calls/drops single-locus outliers along the genome

Description

Calls/drops single-locus outliers along the genome that have a signal that differ significantly from the neighboring loci.

Usage

```
 ## Default S3 method:
callSegmentationOutliers(y, chromosome=0, x=NULL, method="DNAcopy::smooth.CNA", ...,
  verbose=FALSE)
 ## S3 method for class 'data.frame'
callSegmentationOutliers(y, ...)
 ## Default S3 method:
dropSegmentationOutliers(y, ...)
 ## S3 method for class 'data.frame'
dropSegmentationOutliers(y, ...)
```

Arguments

y	A numeric vector of J genomic signals to be segmented.
chromosome	(Optional) An integer scalar (or a vector of length J contain a unique value). Only used for annotation purposes.
x	Optional numeric vector of J genomic locations. If NULL, index locations 1:J are used.
method	A character string specifying the method used for calling outliers.
...	Additional arguments passed to internal outlier detection method.
verbose	See Verbose.

Value

callSegmentationOutliers() returns a logical vector of length J. dropSegmentationOutliers() returns an object of the same type as argument y, where the signals for which outliers were called have been set to NA.

Missing and non-finite values

Signals as well as genomic positions may contain missing values, i.e. NAs or NaNs. By definition, these cannot be outliers.

Author(s)

Henrik Bengtsson

See Also

Internally smooth.CNA is utilized to identify the outliers.

CBS *The CBS class*

Description

A CBS object holds results from the Circular Binary Segmentation (CBS) method for *one* sample for one or more chromosomes.

Package: PSCBS
Class CBS

```
list
~~|
~~+--AbstractCBS
~~~~~~~|
~~~~~~~+--CBS
```

Directly known subclasses:

public abstract static class **CBS**
extends *AbstractCBS*

Usage

```
CBS(...)
```

Arguments

`...`	Arguments passed to the constructor of `AbstractCBS`.

Fields and Methods

Methods:

`append`	-
`as`	-
`estimateStandardDeviation`	-
`plotTracks`	-
`pruneBySdUndo`	-
`segmentByCBS`	-
`seqOfSegmentsByDP`	-
`writeSegments`	-

Methods inherited from AbstractCBS:
adjustPloidyScale, all.equal, append, as.data.frame, clearCalls, drawChangePoints, drawKnownSegments, dropChangePoint, dropChangePoints, dropRegion, dropRegions, extractCNs, extractChromosome, extractChromosomes, extractRegions, extractSegments, extractWIG, getChangePoints, getChromosomeOffsets, getChromosomeRanges, getChromosomes, getLocusData, getLocusSignalNames, getMeanEstimators, getSampleName, getSegmentSizes, getSegmentTrackPrefixes, getSegments, load, mergeThreeSegments, mergeTwoSegments, nbrOfChangePoints, nbrOfChromosomes, nbrOfLoci, nbrOfSegments, normalizeTotalCNs, ploidy, ploidy<-, plotTracks, print, pruneByDP, pruneByHClust, renameChromosomes, report, resegment, resetSegments, sampleCNs, sampleName, sampleName<-, save, seqOfSegmentsByDP, setLocusData, setMeanEstimators, setPloidy, setSampleName, setSegments, shiftTCN, tileChromosomes, updateMeans, writeWIG

Methods inherited from list:
all.equal, as.data.frame, attachLocally, callHooks, relist, within

Difference to DNAcopy object

A CBS object is similar to DNAcopy objects with the major difference that a CBS object holds only one sample, whereas a DNAcopy object can hold more than one sample.

See also

The segmentByCBS() method returns an object of this class.

Author(s)

Henrik Bengtsson

---

findLargeGaps *Identifies gaps of a genome where there exist no observations*

Description

Identifies gaps of a genome where there exist no observations.

Usage

```
## Default S3 method:
findLargeGaps(chromosome=NULL, x, minLength, resolution=1L, ...)
```

Arguments

chromosome	(Optional) An integer vector of length J of chromosome indices.
x	A numeric vector of J of genomic locations.
minLength	A positive numeric scalar specifying the minimum length of a gap.
resolution	A non-negative numeric specifying the minimum length unit, which by default equals one nucleotide/base pair.
...	Not used.

Value

Returns data.frame zero or more rows and with columns chromosome (if given), start, stop, and length.

Author(s)

Henrik Bengtsson

See Also

Use gapsToSegments() to turn the set of identified gaps into the complementary set of segments such that they can be passed to segmentByCBS(), segmentByPairedPSCBS() and segmentByNonPairedPSCBS() via argument knownSegments.

gapsToSegments.data.frame *Gets the genomic segments that are complementary to the gaps*

Description

Gets the genomic segments that are complementary to the gaps, with default chromosome boundaries being `-Inf` and `+Inf`.

Usage

```
## S3 method for class 'data.frame'
gapsToSegments(gaps, resolution=1L, minLength=0L, dropGaps=FALSE, ...)
```

Arguments

`gaps`	A `data.frame` with columns `chromosome`, `start`, and `stop`. Any overlapping gaps will throw an error.
`resolution`	A non-negative `numeric` specifying the minimum length unit, which by default equals one nucleotide/base pair.
`minLength`	Minimum length of segments to be kept.
`dropGaps`	If `TRUE`, the gaps themselves are not part of the output.
`...`	Not used.

Value

Returns `data.frame` of least one row with columns `chromosome` if that argument is given), `start`, `stop` and `length`. The segments are ordered along the genome.

Author(s)

Henrik Bengtsson

See Also

`findLargeGaps`().

NonPairedPSCBS *The NonPairedPSCBS class*

Description

Package: PSCBS
Class NonPairedPSCBS

```
list
~~|
~~+--AbstractCBS
~~~~~~~|
~~~~~~~+--PSCBS
~~~~~~~~~~~~|
~~~~~~~~~~~~+--NonPairedPSCBS
```

Directly known subclasses:

public abstract static class **NonPairedPSCBS**
extends *PSCBS*

A NonPairedPSCBS is an object containing the results from the Non-paired PSCBS method.

Usage

```
NonPairedPSCBS(fit=list(), ...)
```

Arguments

`fit`	A `list` structure containing the Non-paired PSCBS results.
`...`	Not used.

Fields and Methods

Methods:
No methods defined.

Methods inherited from PSCBS:
append, as.data.frame, drawChangePoints, extractChromosomes, extractWIG, getLocusData, getLocusSignalNames, getSegmentTrackPrefixes, isLocallyPhased, isSegmentSplitter, normalizeTotalCNs, writeSegments

Methods inherited from AbstractCBS:
adjustPloidyScale, all.equal, append, as.data.frame, clearCalls, drawChangePoints, drawKnownSegments, dropChangePoint, dropChangePoints, dropRegion, dropRegions, extractCNs, extractChromosome, extractChromosomes, extractRegions, extractSegments, extractWIG, getChangePoints,

getChromosomeOffsets, getChromosomeRanges, getChromosomes, getLocusData, getLocusSignalNames, getMeanEstimators, getSampleName, getSegmentSizes, getSegmentTrackPrefixes, getSegments, load, mergeThreeSegments, mergeTwoSegments, nbrOfChangePoints, nbrOfChromosomes, nbrOfLoci, nbrOfSegments, normalizeTotalCNs, ploidy, ploidy<-, plotTracks, print, pruneByDP, pruneByHClust, renameChromosomes, report, resegment, resetSegments, sampleCNs, sampleName, sampleName<-, save, seqOfSegmentsByDP, setLocusData, setMeanEstimators, setPloidy, setSampleName, setSegments, shiftTCN, tileChromosomes, updateMeans, writeWIG

Methods inherited from list:
all.equal, as.data.frame, attachLocally, callHooks, relist, within

Author(s)

Henrik Bengtsson

See Also

The segmentByNonPairedPSCBS() method returns an object of this class.

`PairedPSCBS` *The PairedPSCBS class*

Description

Package: PSCBS
Class PairedPSCBS

```
list
~~|
~~+--AbstractCBS
~~~~~~~|
~~~~~~~+--PSCBS
~~~~~~~~~~~~|
~~~~~~~~~~~~+--PairedPSCBS
```

Directly known subclasses:

public abstract static class **PairedPSCBS**
extends *PSCBS*

A PairedPSCBS is an object containing the results from the Paired PSCBS method.

Usage

```
PairedPSCBS(fit=list(), ...)
```

Arguments

fit	A list structure containing the Paired PSCBS results.
...	Not used.

Fields and Methods

Methods:

callAB	-
callCopyNeutral	-
callGNL	-
callGNLByTCNofAB	-
callGainNeutralLoss	-
callLOH	-
callNTCN	-
callROH	-
estimateDeltaAB	-
estimateDeltaLOH	-
estimateKappa	-
extractCNs	-
hasBootstrapSummaries	-
plotTracks	-
segmentByNonPairedPSCBS	-
segmentByPairedPSCBS	-
seqOfSegmentsByDP	-

Methods inherited from PSCBS:
append, as.data.frame, drawChangePoints, extractChromosomes, extractWIG, getLocusData, getLocusSignalNames, getSegmentTrackPrefixes, isLocallyPhased, isSegmentSplitter, normalizeTotalCNs, writeSegments

Methods inherited from AbstractCBS:
adjustPloidyScale, all.equal, append, as.data.frame, clearCalls, drawChangePoints, drawKnownSegments, dropChangePoint, dropChangePoints, dropRegion, dropRegions, extractCNs, extractChromosome, extractChromosomes, extractRegions, extractSegments, extractWIG, getChangePoints, getChromosomeOffsets, getChromosomeRanges, getChromosomes, getLocusData, getLocusSignalNames, getMeanEstimators, getSampleName, getSegmentSizes, getSegmentTrackPrefixes, getSegments, load, mergeThreeSegments, mergeTwoSegments, nbrOfChangePoints, nbrOfChromosomes, nbrOfLoci, nbrOfSegments, normalizeTotalCNs, ploidy, ploidy<-, plotTracks, print, pruneByDP, pruneByHClust, renameChromosomes, report, resegment, resetSegments, sampleCNs, sampleName, sampleName<-, save, seqOfSegmentsByDP, setLocusData, setMeanEstimators, setPloidy, setSampleName, setSegments, shiftTCN, tileChromosomes, updateMeans, writeWIG

Methods inherited from list:
all.equal, as.data.frame, attachLocally, callHooks, relist, within

Author(s)

Henrik Bengtsson

See Also

The segmentByPairedPSCBS() method returns an object of this class.

PSCBS *The PSCBS class*

Description

Package: PSCBS
Class PSCBS

```
list
~~|
~~+--AbstractCBS
~~~~~~~|
~~~~~~~+--PSCBS
```

Directly known subclasses:
NonPairedPSCBS, *PairedPSCBS*

public abstract static class **PSCBS**
extends *AbstractCBS*

A PSCBS is an object containing results from parent-specific copy-number (PSCN) segmentation.

Usage

```
PSCBS(fit=list(), ...)
```

Arguments

fit	A list structure containing the PSCN segmentation results.
...	Not used.

Fields and Methods

Methods:

append	-
isLocallyPhased	-
normalizeTotalCNs	-
writeSegments	-

Methods inherited from AbstractCBS:
adjustPloidyScale, all.equal, append, as.data.frame, clearCalls, drawChangePoints, drawKnownSegments, dropChangePoint, dropChangePoints, dropRegion, dropRegions, extractCNs, extractChromosome, extractChromosomes, extractRegions, extractSegments, extractWIG, getChangePoints, getChromosomeOffsets, getChromosomeRanges, getChromosomes, getLocusData, getLocusSignalNames, getMeanEstimators, getSampleName, getSegmentSizes, getSegmentTrackPrefixes, getSegments, load, mergeThreeSegments, mergeTwoSegments, nbrOfChangePoints, nbrOfChromosomes, nbrOfLoci, nbrOfSegments, normalizeTotalCNs, ploidy, ploidy<-, plotTracks, print, pruneByDP, pruneByHClust, renameChromosomes, report, resegment, resetSegments, sampleCNs, sampleName, sampleName<-, save, seqOfSegmentsByDP, setLocusData, setMeanEstimators, setPloidy, setSampleName, setSegments, shiftTCN, tileChromosomes, updateMeans, writeWIG

Methods inherited from list:
all.equal, as.data.frame, attachLocally, callHooks, relist, within

Author(s)

Henrik Bengtsson

See Also

PairedPSCBS.

`segmentByCBS` *Segment genomic signals using the CBS method*

Description

Segment genomic signals using the CBS method of the **DNAcopy** package. This is a convenient low-level wrapper for the `DNAcopy::segment()` method. It is intended to be applied to a sample at the time. For more details on the Circular Binary Segmentation (CBS) method see [1,2].

Usage

```
## Default S3 method:
segmentByCBS(y, chromosome=0L, x=NULL, index=seq_along(y), w=NULL, undo=0,
  avg=c("mean", "median"), ..., joinSegments=TRUE, knownSegments=NULL, seed=NULL,
  verbose=FALSE)
```

Arguments

`y`	A `numeric vector` of J genomic signals to be segmented.
`chromosome`	Optional `numeric vector` of length J, specifying the chromosome of each loci. If a scalar, it is expanded to a vector of length J.
`x`	Optional `numeric vector` of J genomic locations. If `NULL`, index locations `1:J` are used.
`index`	An optional `integer vector` of length J specifying the genomewide indices of the loci.

w	Optional numeric vector in [0,1] of J weights.
undo	A non-negative numeric. If greater than zero, then arguments undo.splits="sdundo" and undo.SD=undo are passed to DNAcopy::segment(). In the special case when undo is +Inf, the segmentation result will not contain any changepoints (in addition to what is specified by argument knownSegments).
avg	A character string specifying how to calculating segment mean levels *after* change points have been identified.
...	Additional arguments passed to the DNAcopy::segment() segmentation function.
joinSegments	If TRUE, there are no gaps between neighboring segments. If FALSE, the boundaries of a segment are defined by the support that the loci in the segments provides, i.e. there exist a locus at each end point of each segment. This also means that there is a gap between any neighboring segments, unless the change point is in the middle of multiple loci with the same position. The latter is what DNAcopy::segment() returns.
knownSegments	Optional data.frame specifying *non-overlapping* known segments. These segments must not share loci. See findLargeGaps() and gapsToSegments().
seed	An (optional) integer specifying the random seed to be set before calling the segmentation method. The random seed is set to its original state when exiting. If NULL, it is not set.
verbose	See Verbose.

Details

Internally segment of **DNAcopy** is used to segment the signals. This segmentation method support weighted segmentation.

Value

Returns a CBS object.

Reproducibility

The DNAcopy::segment() implementation of CBS uses approximation through random sampling for some estimates. Because of this, repeated calls using the same signals may result in slightly different results, unless the random seed is set/fixed.

Missing and non-finite values

Signals may contain missing values (NA or NaN), but not infinite values (+/-Inf). Loci with missing-value signals are preserved and keep in the result.

Likewise, genomic positions may contain missing values. However, if they do, such loci are silently excluded before performing the segmentation, and are not kept in the results. The mapping between the input locus-level data and ditto of the result can be inferred from the index column of the locus-level data of the result.

None of the input data may have infinite values, i.e. -Inf or +Inf. If so, an informative error is thrown.

Author(s)

Henrik Bengtsson

References

[1] A.B. Olshen, E.S. Venkatraman (aka Venkatraman E. Seshan), R. Lucito and M. Wigler, *Circular binary segmentation for the analysis of array-based DNA copy number data*, Biostatistics, 2004
[2] E.S. Venkatraman and A.B. Olshen, *A faster circular binary segmentation algorithm for the analysis of array CGH data*, Bioinformatics, 2007

See Also

To segment allele-specific tumor copy-number signals from a tumor *with* a matched normal, see `segmentByPairedPSCBS`(). For the same *without* a matched normal, see `segmentByNonPairedPSCBS`().

It is also possible to prune change points after segmentation (with identical results) using `pruneBySdUndo()`.

Examples

```
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Simulating copy-number data
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
set.seed(0xBEEF)

# Number of loci
J <- 1000

mu <- double(J)
mu[200:300] <- mu[200:300] + 1
mu[350:400] <- NA # centromere
mu[650:800] <- mu[650:800] - 1
eps <- rnorm(J, sd=1/2)
y <- mu + eps
x <- sort(runif(length(y), max=length(y))) * 1e5
w <- runif(J)
w[650:800] <- 0.001


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Segmentation
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
fit <- segmentByCBS(y, x=x)
print(fit)
plotTracks(fit)



xlab <- "Position (Mb)"
ylim <- c(-3,3)
xMb <- x/1e6
```

```
plot(xMb,y, pch=20, col="#aaaaaa", xlab=xlab, ylim=ylim)
drawLevels(fit, col="red", lwd=2, xScale=1e-6)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# TESTS
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
fit <- segmentByCBS(y, x=x, seed=0xBEEF)
print(fit)
##   id chromosome       start      end nbrOfLoci    mean
## 1  y          0    55167.82 20774251       201  0.0164
## 2  y          0 20774250.85 29320105        99  1.0474
## 3  y          0 29320104.86 65874675       349 -0.0227
## 4  y          0 65874675.06 81348129       151 -1.0813
## 5  y          0 81348129.20 99910827       200 -0.0612


# Test #1: Reverse the ordering and segment
fitR <- segmentByCBS(rev(y), x=rev(x), seed=0xBEEF)
# Sanity check
stopifnot(all.equal(getSegments(fitR), getSegments(fit)))
# Sanity check
stopifnot(all.equal(rev(getLocusData(fitR)$index), getLocusData(fit)$index))

# Test #2: Reverse, but preserve ordering of 'data' object
fitRP <- segmentByCBS(rev(y), x=rev(x), preserveOrder=TRUE)
stopifnot(all.equal(getSegments(fitRP), getSegments(fit)))


# (Test #3: Change points inbetween data points at the same locus)
x[650:654] <- x[649]
fitC <- segmentByCBS(rev(y), x=rev(x), preserveOrder=TRUE, seed=0xBEEF)

# Test #4: Allow for some missing values in signals
y[450] <- NA
fitD <- segmentByCBS(y, x=x, seed=0xBEEF)


# Test #5: Allow for some missing genomic annotations
x[495] <- NA
fitE <- segmentByCBS(y, x=x, seed=0xBEEF)


# Test #6: Undo all change points found
fitF <- segmentByCBS(y, x=x, undo=Inf, seed=0xBEEF)
print(fitF)
stopifnot(nbrOfSegments(fitF) == 1L)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# MISC.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Emulate a centromere
```

```
x[650:699] <- NA
fit <- segmentByCBS(y, x=x, seed=0xBEEF)
xMb <- x/1e6
plot(xMb,y, pch=20, col="#aaaaaa", xlab=xlab, ylim=ylim)
drawLevels(fit, col="red", lwd=2, xScale=1e-6)

fitC <- segmentByCBS(y, x=x, joinSegments=FALSE, seed=0xBEEF)
drawLevels(fitC, col="blue", lwd=2, xScale=1e-6)



# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Multiple chromosomes
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Appending CBS results
fit1 <- segmentByCBS(y, chromosome=1, x=x)
fit2 <- segmentByCBS(y, chromosome=2, x=x)
fit <- append(fit1, fit2)
print(fit)
plotTracks(fit, subset=NULL, lwd=2, Clim=c(-3,3))


# Segmenting multiple chromosomes at once
chromosomeWG <- rep(1:2, each=J)
xWG <- rep(x, times=2)
yWG <- rep(y, times=2)
fitWG <- segmentByCBS(yWG, chromosome=chromosomeWG, x=xWG)
print(fitWG)
plotTracks(fitWG, subset=NULL, lwd=2, Clim=c(-3,3))

# Assert same results
fit$data[,"index"] <- getLocusData(fitWG)[,"index"] # Ignore 'index'
stopifnot(all.equal(getLocusData(fitWG), getLocusData(fit)))
stopifnot(all.equal(getSegments(fitWG), getSegments(fit)))
```

`segmentByNonPairedPSCBS` *Segment total copy numbers and allele B fractions using the Non-paired PSCBS method*

Description

Segment total copy numbers and allele B fractions using the Non-paired PSCBS method [1]. This method does not requires matched normals. This is a low-level segmentation method. It is intended to be applied to one tumor sample at the time.

Usage

```
## Default S3 method:
```

```
segmentByNonPairedPSCBS(CT, betaT, ..., flavor=c("tcn", "tcn&dh", "tcn,dh",
  "sqrt(tcn),dh", "sqrt(tcn)&dh"), tauA=NA, tauB=1 - tauA, verbose=FALSE)
```

Arguments

CT	A numeric vector of J tumor total copy number (TCN) ratios in [0,+Inf) (due to noise, small negative values are also allowed). The TCN ratios are typically scaled such that copy-neutral diploid loci have a mean of two.
betaT	A numeric vector of J tumor allele B fractions (BAFs) in [0,1] (due to noise, values may be slightly outside as well) or NA for non-polymorphic loci.
...	Additional arguments passed to segmentByPairedPSCBS().
flavor	A character specifying what type of segmentation and calling algorithm to be used.
tauA, tauB	Lower and upper thresholds (tauA < tauB for calling SNPs heterozygous based on the tumor allele B fractions (betaT). If NA, then they are estimates from data.
verbose	See Verbose.

Details

Internally segmentByPairedPSCBS() is used for segmentation. This segmentation method does *not* support weights.

Value

Returns the segmentation results as a NonPairedPSCBS object.

Reproducibility

The "DNAcopy::segment" implementation of CBS uses approximation through random sampling for some estimates. Because of this, repeated calls using the same signals may result in slightly different results, unless the random seed is set/fixed.

Whole-genome segmentation is preferred

Although it is possible to segment each chromosome independently using Paired PSCBS, we strongly recommend to segment whole-genome (TCN,BAF) data at once. The reason for this is that downstream CN-state calling methods, such as the AB and the LOH callers, performs much better on whole-genome data. In fact, they may fail to provide valid calls if done chromsome by chromosome.

Missing and non-finite values

The total copy number signals as well as any optional positions must not contain missing values, i.e. NAs or NaNs. If there are any, an informative error is thrown. Allele B fractions may contain missing values, because such are interpreted as representing non-polymorphic loci.

None of the input signals may have infinite values, i.e. -Inf or +Inf. If so, an informative error is thrown.

Non-Paired PSCBS with known genotypes

If allele B fractions for the matched normal (`betaN`) are not available, but genotypes (`muN`) are, then it is possible to run Paired PSCBS. See `segmentByPairedPSCBS()` for details.

Author(s)

Henrik Bengtsson

References

[1] A.B. Olshen, H. Bengtsson, P. Neuvial, P.T. Spellman, R.A. Olshen, V.E. Seshan, *Parent-specific copy number in paired tumor-normal studies using circular binary segmentation*, Bioinformatics, 2011
[2] H. Bengtsson, P. Neuvial and T.P. Speed, *TumorBoost: Normalization of allele-specific tumor copy numbers from a single pair of tumor-normal genotyping microarrays*, BMC Bioinformatics, 2010

See Also

To segment paired tumor-normal total copy numbers and allele B fractions, see `segmentByPairedPSCBS()`.

To segment total copy numbers, or any other unimodal signals, see `segmentByCBS()`.

Examples

```
verbose <- R.utils::Arguments$getVerbose(-10*interactive(), timestamp=TRUE)

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Load SNP microarray data
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
data <- PSCBS::exampleData("paired.chr01")
str(data)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Paired PSCBS segmentation
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Drop single-locus outliers
dataS <- dropSegmentationOutliers(data)

# Speed up example by segmenting fewer loci
dataS <- dataS[seq(from=1, to=nrow(data), by=20),]

str(dataS)

R.oo::attachLocally(dataS)

# Non-Paired PSCBS segmentation
fit <- segmentByNonPairedPSCBS(CT, betaT=betaT,
                               chromosome=chromosome, x=x,
                               seed=0xBEEF, verbose=verbose)
```

```
print(fit)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Bootstrap segment level estimates
# (used by the AB caller, which, if skipped here,
#  will do it automatically)
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
fit <- bootstrapTCNandDHByRegion(fit, B=100, verbose=verbose)
print(fit)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Calling segments in allelic balance (AB)
# NOTE: Ideally, this should be done on whole-genome data
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Explicitly estimate the threshold in DH for calling AB
# (which be done by default by the caller, if skipped here)
deltaAB <- estimateDeltaAB(fit, flavor="qq(DH)", verbose=verbose)
print(deltaAB)

fit <- callAB(fit, delta=deltaAB, verbose=verbose)
print(fit)


# Even if not explicitly specified, the estimated
# threshold parameter is returned by the caller
stopifnot(fit$params$deltaAB == deltaAB)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Calling segments in loss-of-heterozygosity (LOH)
# NOTE: Ideally, this should be done on whole-genome data
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Explicitly estimate the threshold in C1 for calling LOH
# (which be done by default by the caller, if skipped here)
deltaLOH <- estimateDeltaLOH(fit, flavor="minC1|nonAB", verbose=verbose)
print(deltaLOH)

fit <- callLOH(fit, delta=deltaLOH, verbose=verbose)
print(fit)
plotTracks(fit)

# Even if not explicitly specified, the estimated
# threshold parameter is returned by the caller
stopifnot(fit$params$deltaLOH == deltaLOH)
```

`segmentByPairedPSCBS` *Segment total copy numbers and allele B fractions using the Paired PSCBS method*

Description

Segment total copy numbers and allele B fractions using the Paired PSCBS method [1]. This method requires matched normals. This is a low-level segmentation method. It is intended to be applied to one tumor-normal sample at the time.

Usage

```
## Default S3 method:
segmentByPairedPSCBS(CT, thetaT=NULL, thetaN=NULL, betaT=NULL, betaN=NULL, muN=NULL,
  rho=NULL, chromosome=0, x=NULL, alphaTCN=0.009, alphaDH=0.001, undoTCN=0, undoDH=0,
  ..., avgTCN=c("mean", "median"), avgDH=c("mean", "median"),
  flavor=c("tcn&dh", "tcn,dh", "sqrt(tcn),dh", "sqrt(tcn)&dh", "tcn"), tbn=is.null(rho),
  preserveScale=getOption("PSCBS/preserveScale", FALSE), joinSegments=TRUE,
  knownSegments=NULL, dropMissingCT=TRUE, seed=NULL, verbose=FALSE)
```

Arguments

CT	A numeric vector of J tumor total copy number (TCN) ratios in [0,+Inf) (due to noise, small negative values are also allowed). The TCN ratios are typically scaled such that copy-neutral diploid loci have a mean of two.
thetaT, thetaN	(alternative) As an alternative to specifying tumor TCN *ratios* relative to the match normal by argument CT, on may specify total tumor and normal signals seperately, in which case the TCN ratios CT are calculated as $CT = 2 * thetaT/thetaN$.
betaT	A numeric vector of J tumor allele B fractions (BAFs) in [0,1] (due to noise, values may be slightly outside as well) or NA for non-polymorphic loci.
betaN	A numeric vector of J matched normal BAFs in [0,1] (due to noise, values may be slightly outside as well) or NA for non-polymorphic loci.
muN	An optional numeric vector of J genotype calls in {0,1/2,1} for AA, AB, and BB, respectively, and NA for non-polymorphic loci. If not given, they are estimated from the normal BAFs using callNaiveGenotypes as described in [2].
rho	(alternative to betaT and betaN/muN) A numeric vector of J decrease-of-heterozygosity signals (DHs) in [0,1] (due to noise, values may be slightly larger than one as well). By definition, DH should be NA for homozygous loci and for non-polymorphic loci.
chromosome	(Optional) An integer scalar (or a vector of length J), which can be used to specify which chromosome each locus belongs to in case multiple chromosomes are segments. This argument is also used for annotation purposes.
x	Optional numeric vector of J genomic locations. If NULL, index locations 1:J are used.
alphaTCN, alphaDH	The significance levels for segmenting total copy numbers (TCNs) and decrease-in-heterozygosity signals (DHs), respectively.
undoTCN, undoDH	Non-negative numerics. If greater than 0, then a cleanup of segmentions post segmentation is done. See argument undo of segmentByCBS() for more details.

avgTCN, avgDH	A character string specifying how to calculating segment mean levels *after* change points have been identified.
...	Additional arguments passed to segmentByCBS().
flavor	A character specifying what type of segmentation and calling algorithm to be used.
tbn	If TRUE, betaT is normalized before segmentation using the TumorBoost method [2], otherwise not.
preserveScale	Passed to normalizeTumorBoost, which is only called if tbn is TRUE.
joinSegments	If TRUE, there are no gaps between neighboring segments. If FALSE, the boundaries of a segment are defined by the support that the loci in the segments provides, i.e. there exist a locus at each end point of each segment. This also means that there is a gap between any neighboring segments, unless the change point is in the middle of multiple loci with the same position. The latter is what DNAcopy::segment() returns.
knownSegments	Optional data.frame specifying *non-overlapping* known segments. These segments must not share loci. See findLargeGaps() and gapsToSegments().
dropMissingCT	If TRUE, loci for which 'CT' is missing are dropped, otherwise not.
seed	An (optional) integer specifying the random seed to be set before calling the segmentation method. The random seed is set to its original state when exiting. If NULL, it is not set.
verbose	See Verbose.

Details

Internally segmentByCBS() is used for segmentation. The Paired PSCBS segmentation method does *not* support weights.

Value

Returns the segmentation results as a PairedPSCBS object.

Reproducibility

The "DNAcopy::segment" implementation of CBS uses approximation through random sampling for some estimates. Because of this, repeated calls using the same signals may result in slightly different results, unless the random seed is set/fixed.

Whole-genome segmentation is preferred

Although it is possible to segment each chromosome independently using Paired PSCBS, we strongly recommend to segment whole-genome (TCN,BAF) data at once. The reason for this is that downstream CN-state calling methods, such as the AB and the LOH callers, performs much better on whole-genome data. In fact, they may fail to provide valid calls if done chromsome by chromosome.

Missing and non-finite values

The total copy number signals as well as any optional positions must not contain missing values, i.e. NAs or NaNs. If there are any, an informative error is thrown. Allele B fractions may contain missing values, because such are interpreted as representing non-polymorphic loci.

None of the input signals may have infinite values, i.e. -Inf or +Inf. If so, an informative error is thrown.

Paired PSCBS with only genotypes

If allele B fractions for the matched normal (betaN) are not available, but genotypes (muN) are, then it is possible to run a version of Paired PSCBS where TumorBoost normalization of the tumor allele B fractions is skipped. In order for this to work, argument tbn must be set to FALSE.

Author(s)

Henrik Bengtsson

References

[1] A.B. Olshen, H. Bengtsson, P. Neuvial, P.T. Spellman, R.A. Olshen, V.E. Seshan, *Parent-specific copy number in paired tumor-normal studies using circular binary segmentation*, Bioinformatics, 2011
[2] H. Bengtsson, P. Neuvial and T.P. Speed, *TumorBoost: Normalization of allele-specific tumor copy numbers from a single pair of tumor-normal genotyping microarrays*, BMC Bioinformatics, 2010

See Also

Internally, callNaiveGenotypes is used to call naive genotypes, normalizeTumorBoost is used for TumorBoost normalization, and segmentByCBS() is used to segment TCN and DH separately.

To segment tumor total copy numbers and allele B fractions *without* a matched normal, see segmentByNonPairedPSCBS().

To segment total copy-numbers, or any other unimodal signals, see segmentByCBS().

Examples

```
verbose <- R.utils::Arguments$getVerbose(-10*interactive(), timestamp=TRUE)

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Load SNP microarray data
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
data <- PSCBS::exampleData("paired.chr01")
str(data)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Paired PSCBS segmentation
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Drop single-locus outliers
```

```
dataS <- dropSegmentationOutliers(data)

# Speed up example by segmenting fewer loci
dataS <- dataS[seq(from=1, to=nrow(data), by=10),]

str(dataS)

R.oo::attachLocally(dataS)

# Paired PSCBS segmentation
fit <- segmentByPairedPSCBS(CT, betaT=betaT, betaN=betaN,
                            chromosome=chromosome, x=x,
                            seed=0xBEEF, verbose=verbose)
print(fit)

# Plot results
plotTracks(fit)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Bootstrap segment level estimates
# (used by the AB caller, which, if skipped here,
#  will do it automatically)
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
fit <- bootstrapTCNandDHByRegion(fit, B=100, verbose=verbose)
print(fit)
plotTracks(fit)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Calling segments in allelic balance (AB)
# NOTE: Ideally, this should be done on whole-genome data
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Explicitly estimate the threshold in DH for calling AB
# (which be done by default by the caller, if skipped here)
deltaAB <- estimateDeltaAB(fit, flavor="qq(DH)", verbose=verbose)
print(deltaAB)
## [1] 0.1657131

fit <- callAB(fit, delta=deltaAB, verbose=verbose)
print(fit)
plotTracks(fit)

# Even if not explicitly specified, the estimated
# threshold parameter is returned by the caller
stopifnot(fit$params$deltaAB == deltaAB)


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Calling segments in loss-of-heterozygosity (LOH)
# NOTE: Ideally, this should be done on whole-genome data
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Explicitly estimate the threshold in C1 for calling LOH
```

```
# (which be done by default by the caller, if skipped here)
deltaLOH <- estimateDeltaLOH(fit, flavor="minC1|nonAB", verbose=verbose)
print(deltaLOH)
## [1] 0.625175

fit <- callLOH(fit, delta=deltaLOH, verbose=verbose)
print(fit)
plotTracks(fit)

# Even if not explicitly specified, the estimated
# threshold parameter is returned by the caller
stopifnot(fit$params$deltaLOH == deltaLOH)
```

Index

∗Topic **IO**
  callSegmentationOutliers, 3
  findLargeGaps, 6
  gapsToSegments.data.frame, 7
  segmentByCBS, 12
  segmentByNonPairedPSCBS, 16
  segmentByPairedPSCBS, 19
∗Topic **classes**
  CBS, 4
  NonPairedPSCBS, 8
  PairedPSCBS, 9
  PSCBS, 11
∗Topic **methods**
  callSegmentationOutliers, 3
  gapsToSegments.data.frame, 7
∗Topic **package**
  PSCBS-package, 2

AbstractCBS, *4, 5*, *8, 9*, *11*

callNaiveGenotypes, *20*, *22*
callSegmentationOutliers, 3
CBS, 4, *13*
character, *4*, *13*, *17*, *21*

data.frame, *6, 7*, *13*, *21*
dropSegmentationOutliers
        *(*callSegmentationOutliers*)*, 3

FALSE, *13*, *21, 22*
findLargeGaps, 6, *7*, *13*, *21*

gapsToSegments, *6*, *13*, *21*
gapsToSegments
        *(*gapsToSegments.data.frame*)*, 7
gapsToSegments.data.frame, 7

Inf, *13*, *17*, *20*, *22*
integer, *4*, *6*, *12, 13*, *20, 21*

list, *8*, *10, 11*

logical, *4*

NA, *4*, *13*, *17*, *20*, *22*
NaN, *4*, *13*, *17*, *22*
NonPairedPSCBS, 8, *11*, *17*
normalizeTumorBoost, *21, 22*
NULL, *4*, *12, 13*, *20, 21*
numeric, *4*, *6, 7*, *12, 13*, *17*, *20*

PairedPSCBS, 9, *11, 12*, *21*
pruneBySdUndo, *14*
PSCBS, *8, 9*, 11
PSCBS-package, 2

segment, *13*
segmentByCBS, *2*, *6*, 12, *18*, *20–22*
segmentByNonPairedPSCBS, *2*, *6*, *9*, *14*, 16, *22*
segmentByPairedPSCBS, *2*, *6*, *11*, *14*, *17, 18*, 19
smooth.CNA, *4*

TRUE, *7*, *13*, *21*

vector, *4*, *6*, *12, 13*, *17*, *20*
Verbose, *4*, *13*, *17*, *21*