Package: CNVMetrics (via r-universe)

October 17, 2026

Type Package

Version 1.17.0

Date 2021-11-23

Title Copy Number Variant Metrics

Description The CNVMetrics package calculates similarity metrics to facilitate copy number variant comparison among samples and/or methods. Similarity metrics can be employed to compare CNV profiles of genetically unrelated samples as well as those with a common genetic background. Some metrics are based on the shared amplified/deleted regions while other metrics rely on the level of amplification/deletion. The data type used as input is a plain text file containing the genomic position of the copy number variations, as well as the status and/or the log2 ratio values. Finally, a visualization tool is provided to explore resulting metrics.

Encoding UTF-8

License Artistic-2.0

Depends R (>= 4.0)

Imports GenomicRanges, IRanges, S4Vectors, BiocParallel, methods, magrittr, stats, pheatmap, gridExtra, grDevices, rBeta2009

Suggests BiocStyle, knitr, rmarkdown, testthat, XVector

biocViews BiologicalQuestion, Software, CopyNumberVariation

VignetteBuilder knitr

URL https://github.com/krasnitzlab/CNVMetrics, https://krasnitzlab.github.io/CNVMetrics/

BugReports https://github.com/krasnitzlab/CNVMetrics/issues

RoxygenNote 7.3.3

Repository https://bioc.r-universe.dev

Date/Publication 2026-04-28 12:57:34 UTC

RemoteUrl https://github.com/bioc/CNVMetrics

RemoteRef HEAD

RemoteSha 39112e8a1955ccd1c9a9080094a2c50078cd0b99

Contents

CNVMetrics-package 2
calculateLog2ratioMetric 3
calculateOverlapMetric 5
is.CNVMetric 7
plotMetric 8
print.CNVMetric 9
processSim 10

Index **12**

`CNVMetrics-package` *CNVMetrics: Copy number variant metrics*

Description

The CNVMetrics package calculates similarity metrics to facilitate copy number variant comparison among samples and/or methods. Similarity metrics can be employed to compare CNV profiles of genetically unrelated samples as well as those with a common genetic background. Some metrics are based on the shared amplified/deleted regions while other metrics rely on the level of amplification/deletion. The data type used as input is a plain text file containing the genomic position of the copy number variations, as well as the status and/or the log2 ratio values. Finally, a visualization tool is provided to explore resulting metrics.

Author(s)

Astrid Deschênes, Pascal Belleau, David A. Tuveson and Alexander Krasnitz

Maintainer: Astrid Deschênes <adeschen@hotmail.com>

See Also

- `calculateOverlapMetric` for calculating metric using overlapping amplified/deleted regions
- `calculateLog2ratioMetric` for calculating metric using log2ratio values
- `processSim` for generating simulations
- `plotMetric` for plotting metrics

`calculateLog2ratioMetric` *Calculate metric using overlapping amplified/deleted regions*

Description

This function calculates a specific metric, as specified by the user, using overlapping amplified/deleted regions between to samples. The metric is calculated for the amplified and deleted regions separately. When more than 2 samples are present, the metric is calculated for each sample pair.

Usage

```
calculateLog2ratioMetric(
  segmentData,
  method = c("weightedEuclideanDistance"),
  minThreshold = 0.2,
  excludedRegions = NULL,
  nJobs = 1
)
```

Arguments

`segmentData`	a `GRangesList` that contains a collection of genomic ranges representing copy number events, including amplified/deleted status, from at least 2 samples. All samples must have a metadata column called '`log2ratio`' with the log2ratio values.
`method`	a `character` string representing the metric to be used. This should be (an unambiguous abbreviation of) one of "weightedEuclideanDistance". Default: "weightedEuclideanDistance".
`minThreshold`	a single positive `numeric` setting the minimum value to consider two segments as different during the metric calculation. If the absolute difference is below or equal to threshold, the difference will be replaced by zero. Default: 0.2.
`excludedRegions`	an optional `GRanges` containing the regions that have to be excluded for the metric calculation. Default: `NULL`.
`nJobs`	a single positive `integer` specifying the number of worker jobs to create in case of distributed computation. Default: `1` and always `1` for Windows.

Details

The weighted euclidean distance is $(\sum((x_i - y_i)^2 * log(nbrBases_i))^0.5$ where `x` and `y` are the values of 2 samples for a specific segment `i` and `nbrBases` the number of bases of the segment `i`.

Value

an object of class "CNVMetric" which contains the calculated metric. This object is a list with the following components:

- LOG2RATIO a lower-triangular matrix with the results of the selected metric on the log2ratio values for each paired samples. The value NA is present when the metric cannot be calculated. The value NA is also present in the top-triangular section, as well as the diagonal, of the matrix.

The object has the following attributes (besides "class" equal to "CNVMetric"):

- metric the metric used for the calculation.
- names the names of the two matrix containing the metrics for the amplified and deleted regions.

Author(s)

Astrid Deschênes, Pascal Belleau

Examples

```
## Load required package to generate the samples
require(GenomicRanges)

## Create a GRangesList object with 3 samples
## The stand of the regions doesn't affect the calculation of the metric
demo <- GRangesList()
demo[["sample01"]] <- GRanges(seqnames="chr1",
    ranges=IRanges(start=c(1905048, 4554832, 31686841),
    end=c(2004603, 4577608, 31695808)), strand="*",
    log2ratio=c(2.5555, 1.9932, -0.9999))

demo[["sample02"]] <- GRanges(seqnames="chr1",
    ranges=IRanges(start=c(1995066, 31611222, 31690000),
    end=c(2204505, 31689898, 31895666)), strand=c("-", "+", "+"),
    log2ratio=c(0.3422, 0.5454, -1.4444))

## The amplified region in sample03 is a subset of the amplified regions
## in sample01
demo[["sample03"]] <- GRanges(seqnames="chr1",
    ranges=IRanges(start=c(1906069, 4558838),
    end=c(1909505, 4570601)), strand="*",
    log2ratio=c(3.2222, -1.3232))

## Calculating Sorensen metric
calculateLog2ratioMetric(demo, method="weightedEuclideanDistance", nJobs=1)
```

`calculateOverlapMetric` *Calculate metric using overlapping amplified/deleted regions*

Description

This function calculates a specific metric, as specified by the user, using overlapping regions of specific state between to samples. The metric is calculated for each state separately. When more than 2 samples are present, the metric is calculated for each sample pair. By default, the function is calculating metrics for the AMPLIFICATION and DELETION states. However, the user can specify the list of states to be analyzed.

Usage

```
calculateOverlapMetric(
  segmentData,
  states = c("AMPLIFICATION", "DELETION"),
  method = c("sorensen", "szymkiewicz", "jaccard"),
  nJobs = 1
)
```

Arguments

`segmentData`	a `GRangesList` that contains a collection of genomic ranges representing copy number events, including amplified/deleted status, from at least 2 samples. All samples must have a metadata column called '`state`' with a state, in an character string format, specified for each region (ex: DELETION, LOH, AMPLIFICATION, NEUTRAL, etc.).
`states`	a `vector` of `character` string with at least one entry. The strings are representing the states that will be analyzed. Default: c('`AMPLIFICATION`', '`DELETION`').
`method`	a `character` string representing the metric to be used. This should be (an unambiguous abbreviation of) one of "sorensen", "szymkiewicz" or "jaccard". Default: "sorensen".
`nJobs`	a single positive `integer` specifying the number of worker jobs to create in case of distributed computation. Default: `1` and always `1` for Windows.

Details

The two methods each estimate the overlap between paired samples. They use different metrics, all in the range [0, 1] with 0 indicating no overlap. The `NA` is used when the metric cannot be calculated.

The available metrics are (written for two GRanges):

`sorensen`:

This metric is calculated by dividing twice the size of the intersection by the sum of the size of the two sets. With this metric, an overlap metric value of 1 is only obtained when the two samples are identical.

szymkiewicz:

This metric is calculated by dividing the size of the intersection by the size of the smallest set. With this metric, if one set is a subset of the other set, the overlap metric value is 1.

jaccard:

This metric is calculated by dividing the size of the intersection by the size of the union of the two sets. With this metric, an overlap metric value of 1 is only obtained when the two samples are identical.

Value

an object of class "CNVMetric" which contains the calculated metric. This object is a list where each entry corresponds to one state specified in the 'states' parameter. Each entry is a matrix:

- state a lower-triangular matrix with the results of the selected metric on the amplified regions for each paired samples. The value NA is present when the metric cannot be calculated. The value NA is also present in the top-triangular section, as well as the diagonal, of the matrix.

The object has the following attributes (besides "class" equal to "CNVMetric"):

- metric the metric used for the calculation.
- names the names of the two matrix containing the metrics for the amplified and deleted regions.

Author(s)

Astrid Deschênes, Pascal Belleau

References

Sørensen, Thorvald. n.d. "A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species and Its Application to Analyses of the Vegetation on Danish Commons." Biologiske Skrifter, no. 5: 1–34.

Vijaymeena, M. K, and Kavitha K. 2016. "A Survey on Similarity Measures in Text Mining." Machine Learning and Applications: An International Journal 3 (1): 19–28. doi: https://doi.org/10.5121/mlaij.2016.3103

Jaccard, P. (1912), The Distribution of the Flora in the Alpine Zone. New Phytologist, 11: 37-50. doi: https://doi.org/10.1111/j.1469-8137.1912.tb05611.x

Examples

```
## Load required package to generate the samples
require(GenomicRanges)

## Create a GRangesList object with 3 samples
## The stand of the regions doesn't affect the calculation of the metric
demo <- GRangesList()
demo[["sample01"]] <- GRanges(seqnames="chr1",
    ranges=IRanges(start=c(1905048, 4554832, 31686841, 32686222),
    end=c(2004603, 4577608, 31695808, 32689222)), strand="*",
```

```
    state=c("AMPLIFICATION", "AMPLIFICATION", "DELETION", "LOH"))

demo[["sample02"]] <- GRanges(seqnames="chr1",
    ranges=IRanges(start=c(1995066, 31611222, 31690000, 32006222),
    end=c(2204505, 31689898, 31895666, 32789233)),
    strand=c("-", "+", "+", "+"),
    state=c("AMPLIFICATION", "AMPLIFICATION", "DELETION", "LOH"))

## The amplified region in sample03 is a subset of the amplified regions
## in sample01
demo[["sample03"]] <- GRanges(seqnames="chr1",
    ranges=IRanges(start=c(1906069, 4558838),
    end=c(1909505, 4570601)), strand="*",
    state=c("AMPLIFICATION", "DELETION"))

## Calculating Sorensen metric for both AMPLIFICATION and DELETION
calculateOverlapMetric(demo, method="sorensen", nJobs=1)

## Calculating Szymkiewicz-Simpson metric on AMPLIFICATION only
calculateOverlapMetric(demo, states="AMPLIFICATION", method="szymkiewicz",
    nJobs=1)

## Calculating Jaccard metric on LOH only
calculateOverlapMetric(demo, states="LOH", method="jaccard", nJobs=1)
```

`is.CNVMetric` *Is an object of class* `CNVMetric`

Description

Functions to test inheritance relationships between an object and class `CNVMetric`.

Usage

```
## S3 method for class 'CNVMetric'
is(object, class2)
```

Arguments

`object`	an object.
`class2`	a `character` string giving the name of the class.

Value

a `logical`.

plotMetric *Plot metrics present in a* CNVMetric *object*

Description

This function plots one heatmap (or two heatmaps) of the metrics present in a CNVMetric object. For the overlapping metrics, the user can select to print the heatmap related to amplified or deleted regions or both. The NA values present in the metric matrix are transformed into zero for the creation of the heatmap.

Usage

```
plotMetric(
  metric,
  type = "ALL",
  colorRange = c(c("white", "darkblue")),
  show_colnames = FALSE,
  silent = TRUE,
  ...
)
```

Arguments

metric	a CNVMetric object containing the metrics calculated by calculateOverlapMetric or by calculateLog2ratioMetric.
type	a single character string indicating which graph to generate. This should be a type present in the CNVMetric object or "ALL". This is useful for the overlapping metrics that have multiple types specified by the user. Default: "ALL".
colorRange	a vector of 2 character string representing the 2 colors that will be assigned to the lowest (0) and highest value (1) in the heatmap. Default: c("white", "darkblue").
show_colnames	a boolean specifying if column names are be shown. Default: FALSE.
silent	a boolean specifying if the plot should not be drawn. Default: TRUE.
...	further arguments passed to pheatmap::pheatmap() method. Beware that the filename argument cannot be used when type is "ALL".

Value

a gtable object containing the heatmap(s) of the specified metric(s).

Author(s)

Astrid Deschênes

See Also

The default method pheatmap::pheatmap().

Examples

```
## Load required package to generate the samples
require(GenomicRanges)

## Create a GRangesList object with 3 samples
## The stand of the regions doesn't affect the calculation of the metric
demo <- GRangesList()
demo[["sample01"]] <- GRanges(seqnames="chr1",
    ranges=IRanges(start=c(1905048, 4554832, 31686841),
    end=c(2004603, 4577608, 31695808)), strand="*",
    state=c("AMPLIFICATION", "AMPLIFICATION", "DELETION"))

demo[["sample02"]] <- GRanges(seqnames="chr1",
    ranges=IRanges(start=c(1995066, 31611222, 31690000),
    end=c(2204505, 31689898, 31895666)), strand=c("-", "+", "+"),
    state=c("AMPLIFICATION", "AMPLIFICATION", "DELETION"))

## The amplified region in sample03 is a subset of the amplified regions
## in sample01
demo[["sample03"]] <- GRanges(seqnames="chr1",
    ranges=IRanges(start=c(1906069, 4558838),
    end=c(1909505, 4570601)), strand="*",
    state=c("AMPLIFICATION", "DELETION"))

## Calculating Sorensen metric
metric <- calculateOverlapMetric(demo, method="sorensen")

## Plot both amplification and deletion metrics
plotMetric(metric, type="ALL")

## Extra parameters, used by pheatmap(), can also be passed to the function
## Here, we have the metric values print to the cell while the
## row names and column names are removed
plotMetric(metric, type="DELETION", show_rownames=FALSE,
    show_colnames=FALSE, main="deletion", display_numbers=TRUE,
    number_format="%.2f")
```

`print.CNVMetric` *Print CNVMetric object*

Description

Print a `CNVMetric` object and returns it `invisibly`.

Usage

```
## S3 method for class 'CNVMetric'
print(x, ...)
```

Arguments

x	the output object from calculateOverlapRegionsMetric function to be printed.
...	further arguments passed to or from other methods.

Value

the argument x.

See Also

The default method print.default.

---

processSim *Generate simulated samples with copy number profiles derived from a specific sample*

Description

The function uses the input sample to simulate new samples. The simulated samples will possess similar sizes of events, proportional to the original chromosome. To generate realistic simulations, the specified sample must contain segments covering the majority of the genome. Most importantly, the NEUTRAL segments should be present.

Usage

```
processSim(curSample, nbSim)
```

Arguments

curSample	a GRanges that contains a collection of genomic ranges representing copy number events, including amplified/deleted status, from one sample. The sample must have a metadata column called 'state' with a state, in an character string format, specified for each region (ex: DELETION, LOH, AMPLIFICATION, NEUTRAL, etc.) and a metadata column called 'CN' that contains the log2 copy number ratios.
nbSim	a single positive integer which is corresponding to the number of simulations that will be generated.

Value

a data.frame containing the segments for each simulated sample. The data.frame has 6 columns:

- ID a character string, the name of the simulated sample
- chr a character string, the name fo the chromosome
- start a integer, the starting position of the segment
- end a integer, the ending position of the segment

- `log2ratio` a `numerical`, the log2 copy number ratio assigned to the segment
- `state` a `character` string, the state of the segment (ex: DELETION, AMPLIFICATION, NEUTRAL, etc.)

Author(s)

Astrid Deschênes, Pascal Belleau

Examples

```
## Load required package to generate the sample
require(GenomicRanges)

## Create one 'demo' genome with 2 chromosomes and few segments
## The stand of the regions doesn't affect the calculation of the metric
sample01 <- GRanges(seqnames=c(rep("chr1", 4), rep("chr2", 3)),
    ranges=IRanges(start=c(1905048, 4554832, 31686841, 32686222,
        1, 120331, 725531),
    end=c(2004603, 4577608, 31695808, 32689222, 117121,
        325555, 1225582)),
    strand="*",
    state=c("AMPLIFICATION", "NEUTRAL", "DELETION", "LOH",
        "DELETION", "NEUTRAL", "NEUTRAL"),
    log2ratio=c(0.5849625, 0, -1, -1, -0.87777, 0, 0))

## Generates 10 simulated genomes based on the 'demo' genome
simRes <- processSim(curSample=sample01, nbSim=10)
```

Index

∗ **package**
CNVMetrics-package, 2

calculateLog2ratioMetric, *2*, 3
calculateOverlapMetric, *2*, 5
CNVMetrics *(*CNVMetrics-package*)*, 2
CNVMetrics-package, 2

is.CNVMetric, 7

pheatmap::pheatmap(), *8*
plotMetric, *2*, 8
print.CNVMetric, 9
print.default, *10*
processSim, *2*, 10