CNTools

April 20, 2011

`CNSeg-class` *Class "CNSeg" contains the output of DNACopy segmentation data that can be operated on by the associated methods*

Description

Segmentation results of copy number data can not be operated by other analysis tools such as classification. The class "CNSeg" provides methods to convert segmentation data into a (what is called Reduced segments (RS))matrix format so that other computations can be performed

Objects from the Class

Objects can be created by calls of the form `new("CNSeg", ...)`. A constructor `CNSeg(segList)` can also be used to instantiate CNSeg object

Slots

`segList`**:** Object of class `"data.frame"` is the "output" element of the object return by the segment function of DNACopy

`chromosome`**:** Object of class `"character"` is the name of the column in segList that contains the chromosome name a given segment belong to

`end`**:** Object of class `"character"` is the name of the column in segList that contains the ending locatino of a given segment

`start`**:** Object of class `"character"` is the name of the column in segList that contains the starting location of a given segment

`segMean`**:** Object of class `"character"` is the name of the column in segList that contains the mean log ratio of a given segment

`id`**:** Object of class `"character"` is the name of the column in segList that contains the id of a given segment

Methods

getRS `signature(object = "CNSeg")`: get the Reduced Segment data matrix

segList `signature(object = "CNSeg")`: get the value for slot "segList" of a CNSeg object

segList<- `signature(object = "CNSeg")`: set the value for slot "segList" of a CNSeg object

show `signature(object = "CNSeg")`: print a CNSeg object

Author(s)

Jianhua Zhang

References

NA

See Also

RS

Examples

```
data("sampleData")
# take a subset of the data for speed
seg <- CNSeg(sampleData[which(is.element(sampleData[, "ID"],
    sample(unique(sampleData[, "ID"]), 10))), ])
rsBypair <- getRS(seg, by = "pair", imput = FALSE, XY = FALSE, what = "mean")
rsBypair
```

`getCor-methods` *Method that extends the cor function of stats to handle reduced segment data*

Description

`getCor` gets data contained by an RS object ready and then calls the cor function for correlation calculations

Methods

x = "ANY" see function cor of stats

x = "RS" method that handles RS objects

`diffBy-methods` *A filter that filters out features that do not differ by a set threshold between a pair of samples*

Description

`madFilter` With a given threshold, the method checks each pair of samples to exclude features that do not differ by the threshold between two samples. This filter is only applicable to pairs of reduced segments

Methods

object = "ANY" not iimplemented yet

object = "RS" method that handles RS objects

`dist-methods` *A method that extends the generic function dist to handle reduced segemnts*

Description

`getDist` gets the data contained by an RS object ready to be passed to the generic functon for distance calculations

Methods

x = "ANY" see dist of stats

x = "RS" takes an RS object and then call the dist function of stats for distance calculations

`genefilter-methods` *A method that filters feature based on reduced segment*

Description

Extends genefilter filtering functions to handle reduced segment data

Methods

expr = "RS" A character string to indicate that filtering is based on reduced segment data

`getRS-methods` *method that convert segment data into reduced segment matrix*

Description

`getRS` takes a CNSeg object containing the output of the segment function of DNAcopy and format the data into a matrix based on overlapping chromosome region (by = region", gene (by = gene) or pair overlapping chromosome region (by = pair)

Methods

object = "CNSeg" a reduced segment can be generated in three ways; by chromosomal regions that overlap across sample (by = region), by genes (by = gene), or by pair of samples with chromosome regions aligned (by = pair). User may choose to imput cells (by region or gene only) where a value can not be assigned by setting imput = TRUE. The X and Y chromosomes can dropped by stting XY = FALSE.

`madFilter-methods` *Method that filters reduced segment matrix by the mean absolute deviation*

Description

`madFilter` calculates the mean absolute deviation across samples for each rows and drops rows that are not above percentile defined by a user

Methods

object = "ANY" not iimplemented yet

object = "RS" method that handles RS objects

`RS-class` *Class "RS" contains the Reduced Segment data matrix derived from the output of segmentation*

Description

The class contains the Reduced Segment data matrix derived from the output of the segment method of DNACopy and provides functions to manipulate the data or perform other computational operations

Objects from the Class

Objects can be created by calls of the form `new("RS", ...)`. A constructor RS(rs, by, imput, XY) can also be used to intantiate an object of this class

Slots

`rs`: Object of class `"ANY"` eith a matrix if the RS is by region or gene or a list of matrix if the RS is by sample pairs

`by`: Object of class `"character"` a charactere string indicating how the RS matrix is obtained. Valid values include region, gene, or pair

Methods

cor `signature(x = "RS")`: Calculates the sample wise correlation coefficients using the Reduced Segment matix

dist `signature(x = "RS")`: Calculates the distance between samples contained in the Reduced Segment matrix

genefilter `signature(expr = "RS")`: filters features by calling the genefilter function

madFilter `signature(object = "RS")`: filters features by mean absolute deviation

rs `signature(object = "RS")`: get method for the by slot of an RS object

rs<- `signature(object = "RS")`: assignment method for the rs slot

segBy `signature(object = "RS")`: get method for the by slot of an RS object

show `signature(object = "RS")`: show method for an RS object

Author(s)

Jianhua Zhang

See Also

`CNSeg`

Examples

```
data("sampleData")
# take a subset of the data for speed
seg <- CNSeg(sampleData[which(is.element(sampleData[, "ID"], sample(unique(sampleData
rsByregion <- getRS(seg, by = "region", imput = TRUE, XY = FALSE, what = "median")
rsByregion
```

Index