snapCGH

November 11, 2009

R topics documented:

`cbind` *Combine SegList Objects*

Description

Combine a series of `SegList` objects.

Usage

```
## S3 method for class 'SegList':
cbind(..., deparse.level=1)
```

Arguments

`...`	`SegList` objects
`deparse.level`	not currently used, see `cbind` in the base package

Details

`cbind` combines data objects assuming the same gene lists but different arrays.

For `cbind`, the matrices of expression data from the individual objects are cbinded. The data.frames of target information, if they exist, are rbinded. The combined data object will preserve any additional components or attributes found in the first object to be combined. It is not recommend to use the is `rbind` function for the `SegList` object. This is because it would require `SegLists` with mutually exclusive chromosomes or would result in combining multiple different segmentations for the same chromosome, which is pointless. If `rbind` is required perform it on an `MAList` and then segment it. It is currently only included as an internal function called within other library functions.

Value

An `SegList` object holding data from all the arrays and all genes from the individual objects.

Author(s)

Gordon Smyth, modified by Mike Smith for SegList object

See Also

`cbind` in the base package.

`chrominfo.Mb` *Basic Chromosomal Information for UCSC Human Genome Assembly July 2003 freeze*

Description

This dataset contains basic chromosomal information for UCSC Human Genome Assembly July 2003 freeze.

Usage

```
chrominfo.basepair
```

Format

A data frame with 24 observations on the following 3 variables.

chrom Chromosomal index, X is coded as 23 and Y as 24.

length Length of each chromosome in megabases.

centromere Location of the centromere on the chromosome (Mb).

Details

This file is used for many plotting functions. The centromeric location is approximately extimated by taking mid-point between the last fish-mapped clone on the p-arm and the first fish-mapped clone on the q-arm using relevant UCSC freeze. For an alternative freeze, one needs to manually create a 3-column file of the format described above.

Source

http://genome.ucsc.edu/cgi-bin/hgText

`compareSegmentations` *Function for comparing segmentation methods to a known truth*

Description

This function takes a SegList and compares the breakpoints indicated in other SegLists with this original one.

Usage

```
compareSegmentations(TrueSeg,offset = 0,...)
```

Arguments

TrueSeg	An object of class SegList which is scored against. Normally the output from simulateData.
offset	Integer value between 0 and 2 specifying how close (in number of clones) to a true breakpoint the segmentation method must be before it is scored.
...	One or more objects of class SegList. These are compared to TrueSeg.

Value

The method returns a list containing two matrices. The first of these, $TPR, contains the true positive rate, whilst the second, $FDR, holds the false discovery rate. Both of these matrices are arranged such that a row represents a segmentation method and each column is an array.

Author(s)

John Marioni and Mike Smith

---

convert.output *Converts the output from the simulation to a format which can be used by segmentation schemes available within R*

Description

This function converts the output obtained by applying our simulation scheme into a format that can be used (either directly or indirectly) as the input to various segmentation schemes available within R. Additionally, we are in the process of submitting a library to CRAN which will enable the user to apply a number of the segmentation schemes available within R to datasets which have the same structure as that generated by this function.

Usage

```
convert.output(input)
```

Arguments

input	The output obtained upon applying the sim.structure function

Details

This function outputs an object which is similar in structure/format to an RG or MA object used in Limma.

Value

This function outputs a list with entries

M	A matrix containing the $\log_2$ ratios
genes	A matrix containing the simulated midpoints and the chromosome which forms the template upon which the simulation is based.

Author(s)

Michael Smith, John Marioni

Examples

```
## The function is currently defined as
function(input){
  holder <- list()
  for (i in 1:length(input)){
  holder[[i]] <- list()}
  for(i in 1:length(input)){
    holder[[i]]$genes <- matrix(NA, nrow = length(input[[i]]$clones$mid.point),
                                ncol = 2)
  }
  for(i in 1:length(input)){
    holder[[i]]$M <- as.matrix(input[[i]]$datamatrix)
    holder[[i]]$genes[,1] <- input[[i]]$clones$mid.point
    holder[[i]]$genes[,2] <- rep(input[[i]]$chrom,length(input[[i]]$clones$mid.point))
    colnames(holder[[i]]$genes) <- c("kb", "Chrom")
    holder[[i]] <- new("aCGHList", holder[[i]])
  }
  holder
  }
```

---

dimnames *Retrieve the Dimension Names of an RGList, MAList or SegList Object*

Description

Retrieve the dimension names of a microarray data object.

Usage

```
## S3 method for class 'SegList':
dimnames(x)
```

Arguments

x	An object of class SegList

Details

The dimensionnames of an microarray object are the same as those of the most important matrix component of that object.

A consequece is that rownames and colnames will work as expected.

Value

Either NULL or a list of length 2. If a list, its components are either 'NULL' or a character vector the length of the appropriate dimension of x.

Author(s)

Gordon Smyth, edited by Mike Smith

See Also

`dimnames` in the base package.

`dim` *Retrieve the Dimensions of an RGList, MAList or SegList Object*

Description

Retrieve the number of rows (genes) and columns (arrays) for an RGList, MAList or SegList object.

Usage

```
## S3 method for class 'SegList':
dim(x)
## S3 method for class 'SegList':
length(x)
```

Arguments

`x`	an object of class `RGList`, `MAList` or `SegList`

Details

Microarray data objects share many analogies with ordinary matrices in which the rows correspond to spots or genes and the columns to arrays. These methods allow one to extract the size of microarray data objects in the same way that one would do for ordinary matrices.

A consequence is that row and column commands `nrow(x)`, `ncol(x)` and so on also work.

Value

Numeric vector of length 2. The first element is the number of rows (genes) and the second is the number of columns (arrays).

Author(s)

Gordon Smyth, modified by Mike Smith for SegList object

See Also

`dim` in the base package.

Examples

```
M <- A <- matrix(11:14,4,2)
rownames(M) <- rownames(A) <- c("a","b","c","d")
colnames(M) <- colnames(A) <- c("A1","A2")
MA <- new("MAList",list(M=M,A=A))
dim(M)
ncol(M)
nrow(M)
length(M)
```

---

filterClones *Filter clones from sample*

Description

Function for filtering clones via a user defined function.

Usage

```
filterClones(MA, filterFunc, ...)
```

Arguments

MA	An object of class MAList
filterFunc	A user specified function that accepts an MAList and returns the indices of the clones to be removed.
...	Additional arguments to be passed to the filter function.

Details

Any clones identified by the filter function are turned into NA's. These are then removed or imputed within the processCGH function.

Author(s)

Mike Smith

---

findBreakPoints *Returns the start and end of segments.*

Description

Function to returns the start and end of segments when given a SegList and an array. Currently only used within the plotSegmentedGenome function.

Usage

```
findBreakPoints(seg, array)
```

Arguments

seg	An object of class SegList.
array	Numeric value corresponding to a column in seg.

Author(s)

Mike Smith

---

find.param.five *Yields the output in a model with five underlying states*

Description

This function is a workhorse of the process.data function. It outputs state means/variances and transitions matrices in the model with five states.

Usage

```
find.param.five(output.optim,var.fixed)
```

Arguments

output.optim	output of the optimisation with 5 underlying states
var.fixed	Logical variable - TRUE if you want to fix the variance to be the same across states

Value

Outputs the mean/variance, transition matrix, maximised likelihood and convergence information

Author(s)

John Marioni

---

find.param.four *Yields output when there are 4 underlying states*

Description

This function provides the output (means/variances, transition matrix, likelihood) when the heterogeneous HMM is fitted with four underlying states. It is a workhorse of the process.data function.

Usage

```
find.param.four(output.optim,var.fixed)
```

Arguments

`output.optim`	The output from fitting a heterogeneous HMM when there are four underlying states
`var.fixed`	Logical variable - TRUE if you want the variance to be tied across states. Defaults to FALSE

Value

This function outputs the state means/variances, transition matrices, rate parameters, maximised likelihood and convergence information provided by fitting a heterogeneous HMM with four underlying states.

Author(s)

John Marioni

`find.param.one` *Yields output when there is 1 underlying states*

Description

This function provides the output (means/variances, transition matrix, likelihood) when the heterogeneous HMM is fitted with one underlying state. It is a workhorse of the process.data function.

Usage

```
find.param.one(output.optim)
```

Arguments

`output.optim`	The output from fitting a heterogeneous HMM when there is one underlying states

Value

This function outputs the state means/variances, transition matrices, rate parameters, maximised likelihood and convergence information provided by fitting a heterogeneous HMM with one underlying state.

Author(s)

John Marioni

find.param.three *Yields output when there are 3 underlying states*

Description

This function provides the output (means/variances, transition matrix, likelihood) when the heterogeneous HMM is fitted with three underlying states. It is a workhorse of the process.data function.

Usage

```
find.param.three(output.optim,var.fixed)
```

Arguments

output.optim	The output from fitting a heterogeneous HMM when there are four underlying states
var.fixed	Logical variable - TRUE if you want the variances to be tied across states. Defaults to FALSE

Value

This function outputs the state means/variances, transition matrices, rate parameters, maximised likelihood and convergence information provided by fitting a heterogeneous HMM with three underlying states.

Author(s)

John Marioni

find.param.two *Yields output when there are 2 underlying states*

Description

This function provides the output (means/variances, transition matrix, likelihood) when the heterogeneous HMM is fitted with two underlying states. It is a workhorse of the process.data function.

Usage

```
find.param.two(output.optim,var.fixed)
```

Arguments

output.optim	The output from fitting a heterogeneous HMM when there are two underlying states
var.fixed	Logical variable - TRUE if you want to tie the variance across states. Defaults to FALSE

Value

This function outputs the state means/variances, transition matrices, rate parameters, maximised likelihood and convergence information provided by fitting a heterogeneous HMM with two underlying states.

Author(s)

John Marioni

`fit.model` *Fitting a heterogneous HMM to the log2 ratios on a particular chromosome.*

Description

This function fits five homogeneous HMMs to the log2 ratios on a particular chromosome. It then uses either the AIC or BIC to determine which of the five models is optimal before using a scaled version of the Viterbi algorithm to assign clones to states with the same underlying copy number.

Usage

```
fit.model(sample, chrom, dat, datainfo = clones.info, useCloneDists = TRUE, cova
aic = TRUE, bic = FALSE, delta = 1, var.fixed=FALSE, epsilon = 1e-06,
numiter = 30000)
```

Arguments

`sample`	If there are multiple samples, the number of the sample to be segmented
`chrom`	The chromosome on which the segmentation is to be carried out on
`dat`	The log2 ratios obtained from the clones located on that chromosome
`datainfo`	A dataframe containing information about the clones on that chromosome (name, chromosome and location (in Mbs))
`useCloneDists`	Boolean stating whether the distance between clones should be incorportated into the HMM. If false then the HMM become homogeneous.
`covariates`	A matrix contianing the covariate information for the clones located on the chromosome to be segmented. It should have length one less than the number of clones as covariate information is not used when segmenting the first clone on the chromosome.
`aic`	Set to true if you want to use the aic. This is the default. Only one of aic and bic should be set to true.
`bic`	Set to true if you want to use the bic.
`delta`	A parameter to be set if you want to use the BIC
`var.fixed`	Logical variable - TRUE if you want to tie the variance to be the same across all states. Defaults to FALSE
`epsilon`	.
`numiter`	Number of iterations to be used in the optimization alogithm.

Value

The output is in the same format as that obtained when the nlm function is applied.

Author(s)

John Marioni and Mike Smith

`genomePlot` *Plots the genome*

Description

Basic plot of the log2 ratios for each array ordered along the genome.

Usage

```
genomePlot(input, array = 1, naut = 22, Y = FALSE, X
                  = FALSE, main = NA, status, values, pch, cex, col, chrominfo =
                  chrominfo.Mb, ylim = c(-2, 2), ylb = "Log2Ratio",
                  chrom.to.plot = NA, xlim = c(0,NA), ...)
```

Arguments

`input`	an object of class `MAList` or `SegList`
`array`	integer of the array (sample) to be plotted.
`naut`	number of autosomes in the organism
`Y`	TRUE if chromosome Y is to be plotted, FALSE otherwise
`X`	TRUE if chromosome X is to be plotted, FALSE otherwise
`main`	Provides the title of the plot
`status`	character vector giving the control status of each spot on the array, of same length as the number of rows of `log2ratios(input)`. If omitted, all points are plotted in the default color, symbol and size.
`values`	character vector giving values of `status` to be highlighted on the plot. Defaults to unique values of `status`. Ignored if there is no `status` vector.
`pch`	vector or list of plotting characters. Default to integer code 16. Ignored is there is no `status` vector.
`col`	numeric or character vector of colors, of the same length as `values`. Defaults to `1:length(values)`. Ignored if there is no `status` vector.
`cex`	numeric vector of plot symbol expansions, of the the same length as `values`. Defaults to 0.2 for the most common `status` value and 1 for the others. Ignored if there is no `status` vector.
`chrominfo`	a chromosomal information associated with the mapping of the data.
`ylim`	Minimum y-scale to use for plotting.
`chrom.to.plot`	Specify which chromosome to plot
`ylb`	label for the Y-axis.
`xlim`	limits for the x-axis
`...`	Any other parameters

Details

The status vector is intended to specify the control status of each spot, for example "gene", "ratio control", "house keeping gene", "buffer" and so on. The vector is usually computed using the function controlStatus and a spot-types file. However the function may be used to highlight any subset of spots.

Author(s)

John Marioni

See Also

MAList SegList

---

heatmapGenome *clustering and heatmap*

Description

This function clusters samples and shows their heatmap

Usage

```
heatmapGenome(input, response = as.factor(rep("All", ncol(input))),
                chrominfo = chrominfo.Mb, cutoff = 1, lowCol =
                "blue", highCol = "yellow", midCol = "white", ncolors =
                50, byclass = FALSE, showaber = FALSE, amplif = 1,
                homdel = -0.75, samplenames = colnames(input),
                vecchrom = 1:22, titles = "Image Plot", methodS =
                "ward", categoricalPheno = TRUE, CENTROMERE = FALSE)
```

Arguments

input	object of class MAList or SegList
response	phenotype of interest. defaults to the same phenotype assigned to all samples
chrominfo	a chromosomal information associated with the mapping of the data
cutoff	maximum absolute value. all the values are floored to +/-cutoff depending on whether they are positive of negative. defaults to 1
ncolors	number of colors in the grid. input to maPalette. defaults to 50
lowCol	color for the low (negative) values. input to maPalette. defaults to "red"
highCol	color for the high (positive) values. input to maPalette. defaults to "green"
midCol	color for the values close to 0. input to maPalette. defaults to "black"
byclass	logical indicating whether samples should be clustered within each level of the phenotype or overall. defaults to F
showaber	logical indicating whether high level amplifications and homozygous deletions should be indicated on the plot. defaults to F
amplif	positive value that all observations equal or exceeding it are marked by yellow dots indicating high-level changes. defaults to 1

homdel	negative value that all observations equal or below it are marked by light blue dots indicating homozygous deletions. defaults to -0.75
samplenames	sample names
vecchrom	vector of chromosomal indeces to use for clustering and to display. defaults to 1:23
titles	plot title. defaults to "Image Plots"
methodS	clustering method to cluster samples. defaults to "ward"
categoricalPheno	logical indicating whether phenotype is categorical. Continious phenotypes are treated as "no groups" except that their values are dispalyed.defaults to TRUE.
CENTROMERE	logical indicating whether to plot the centromere

Details

This functions is a more flexible version of the heatmap. It can cluster within levels of categorical phenotype as well as all of the samples while displaying phenotype levels in different colors. It also uses any combination of chromosomes that is requested and clusters samples based on these chromosomes only. It draws the chromosomal boundaries and displays high level changes and homozygous deletions. If phenotype if not categical, its values may still be displayed but groups are not formed and byclass = F. Image plot has the samples reordered according to clustering order.

See Also

heatmap

IDProbes *Interactive version of genomePlot*

Description

Interactive version of genomePlot. Allows the user to click near a probe and the name of that probe will be displayed next to it.

Usage

```
IDProbes(input, array = 1, naut = 22, Y = FALSE, X
                = FALSE, status, values, pch, cex, col, chrominfo =
                chrominfo.Mb, ylim = c(-2, 2), ylb = "Log2Ratio",
                chrom.to.plot = 1, xlim = c(0,NA))
```

Arguments

input	an object of class MAList or SegList
array	integer of the array (sample) to be plotted.
naut	number of autosomes in the organism
Y	TRUE if chromosome Y is to be plotted, FALSE otherwise
X	TRUE if chromosome X is to be plotted, FALSE otherwise

status	character vector giving the control status of each spot on the array, of same length as the number of rows of log2ratios(input). If omitted, all points are plotted in the default color, symbol and size.
values	character vector giving values of status to be highlighted on the plot. Defaults to unique values of status. Ignored if there is no status vector.
pch	vector or list of plotting characters. Default to integer code 16. Ignored is there is no status vector.
col	numeric or character vector of colors, of the same length as values. Defaults to 1:length(values). Ignored if there is no status vector.
cex	numeric vector of plot symbol expansions, of the the same length as values. Defaults to 0.2 for the most common status value and 1 for the others. Ignored if there is no status vector.
chrominfo	a chromosomal information associated with the mapping of the data.
ylim	Minimum y-scale to use for plotting.
chrom.to.plot	Specify which chromosome to plot
ylb	label for the Y-axis.
xlim	limits for the x-axis

Author(s)

Mike Smith

See Also

genomePlot

---

imputeMissingValues

Imputing log2 ratios

Description

Imputing log2 ratios

Usage

```
imputeMissingValues(seg, chrominfo = chrominfo.Mb, maxChrom =
23, smooth = 0.1)
```

Arguments

seg	Object of class SegList
chrominfo	a chromosomal information associated with the mapping of the data
maxChrom	Highest chromosome to impute
smooth	smoothing parameter for the lowess procedure

Details

There are two main reasons to impute data. One is that given that imputation is reasonable, one can increase the analytical power and improve results. Another, more practical, is that at the moment many widely used fuctions in R do not support missing values. While procedures such as kNN imputations is widely used for gene expression data, it is more powerful to take advantage of the genomic structure of the array CGH data and use a smoother. Note that we perform only one pass of smoothing. If there still remain missing values, they are imputed by the median on the chromosome or chromosomal arm where applicable.

Value

Computes and returns the imputed log2 ratio matrix of the aCGH object.

See Also

`SegList`

`LargeDataObject-class` *Large Data Object - class*

Description

A virtual class including the data classes `RGList`, `MAList` and `SegList`, all of which typically contain large quantities of numerical data in vector, matrices and data.frames.

Methods

A `show` method is defined for objects of class `LargeDataObject` which uses `printHead` to print only the leading elements or rows of components or slots which contain large quantities of data.

Author(s)

Gordon Smyth

Examples

```
#  see normalizeBetweenArrays
```

log2ratios *Extracting log2 ratios*

Description

This function extracts the log2 ratios from either an MAList object or a SegList object.

Usage

```
log2ratios(x)
```

Arguments

x	An object of class MAList or SegList

Author(s)

Mike Smith

mergeStates *Funtion to merge states based on their state means*

Description

mergeStates takes the output of a segementation algorithm in the form of a SegList and iteratively merges the states with means closer than a supplied threshold.

Usage

```
mergeStates(segList, MergeType = 1, pv.thres=0.0001, ansari.sign=0.01, minDiff =
```

Arguments

segList	Object of class SegList.
MergeType	Select either 1 or 2. 1 uses a new merging algorithm developed by Hanni Willenbrock and Jane Fridlyand.
pv.thres	Significance threshold for Wilcoxon test for level merging. Used when MergeType = 1.
ansari.sign	Significance threshold for Ansari-Bradley test. Used when MergeType = 1.
minDiff	The states whose predicted values are less than minDiff apart are merged into one state and all the predicited values are recomputed. Used when MergeType = 2.

Details

This function is intended to reduce effect of the possible small magnitude technological artifacts on the structure determination.

Value

A SegList object is returned with the merged states stored in the pred list.

Author(s)

Jane Fridlyand

References

Application of Hidden Markov Models to the analysis of the array CGH data, Fridlyand et.al., *JMVA*, 2004

See Also

SegList, runHomHMM, runGLAD, runDNAcopy

---

non.zero.length.distr.non.tiled

Empirical distribution of segment lengths in non-tiled regions with copy number gains or losses

---

Description

This file contains the empirical distribution of segment lengths (of untiled regions and whose state mean indicates that they correspond to regions of copy number gain or loss) derived by applying the DNAcopy segmentation scheme (Olshen et al., 2004) to an unpublished breast cancer dataset. Instead of using the physical length of the segments we calculate the lengths as a proportion of the length of the untiled region.

Usage

```
data(non.zero.length.distr.non.tiled)
```

Source

The empirical distribution was derived using an unpublished breast cancer dataset.

---

non.zero.length.distr.tiled

Empirical distribution of segment lengths in tiled regions with copy number gains or losses

---

Description

This file contains the empirical distribution of segment lengths (of tiled regions and whose state mean indicates that they correspond to regions of copy number gain or loss) derived by applying the DNAcopy segmentation scheme (Olshen et al., 2004) to an unpublished breast cancer dataset. Instead of using the physical length of the segments we calculate the lengths as a proportion of the length of the tiled region.

Usage

```
data(non.zero.length.distr.tiled)
```

Source

The empirical distribution was derived using an unpublished breast cancer dataset.

`plotSegmentedGenome` *Plots the genome*

Description

Basic plot of the log2 ratios for each array ordered along the genome.

Usage

```
plotSegmentedGenome(..., array = 1, naut = 22, Y = FALSE, X
                 = FALSE, status, values, pch, cex, col, chrominfo =
                 chrominfo.Mb, ylim = c(-2, 2), ylb = "Log2Ratio",
                 chrom.to.plot = NA, xlim = c(0,NA), colors = NULL,
mark.regions = FALSE, main = NA)
```

Arguments

`...`	Objects of class `SegList`
`array`	integer of the array (sample) to be plotted.
`naut`	number of autosomes in the organism
`Y`	TRUE if chromosome Y is to be plotted, FALSE otherwise
`X`	TRUE if chromosome X is to be plotted, FALSE otherwise
`status`	character vector giving the control status of each spot on the array, of same length as the number of rows of `log2ratios(input)`. If omitted, all points are plotted in the default color, symbol and size.
`values`	character vector giving values of `status` to be highlighted on the plot. Defaults to unique values of `status`. Ignored if there is no `status` vector.
`pch`	vector or list of plotting characters. Default to integer code 16. Ignored is there is no `status` vector.
`col`	numeric or character vector of colors, of the same length as `values`. Defaults to `1:length(values)`. Ignored if there is no `status` vector.
`cex`	numeric vector of plot symbol expansions, of the the same length as `values`. Defaults to 0.2 for the most common `status` value and 1 for the others. Ignored if there is no `status` vector.
`chrominfo`	a chromosomal information associated with the mapping of the data.
`ylim`	Minimum y-scale to use for plotting.
`chrom.to.plot`	Specify which chromosome to plot
`ylb`	label for the Y-axis.

xlim	limits for the x-axis
colors	vector of colors to plot segmented states of each SegList passed to the function.
mark.regions	Boolean. If true will colour code the segmentation plot using the information stored in $regions and generated by nudSegmentation, or an equivalent method.
main	Specify the title of the plot

Details

The status vector is intended to specify the control status of each spot, for example "gene", "ratio control", "house keeping gene", "buffer" and so on. The vector is usually computed using the function controlStatus and a spot-types file. However the function may be used to highlight any subset of spots.

Author(s)

Mike Smith

See Also

genomePlot SegList

---

processCGH *Process data in an MAList*

Description

This function takes object of class MAList and it re-orders and filters clones based on their mapping information and proportion missing. It also average duplicated clones and imputes missing values for clones that are still NA after the filtering step. Note that imputation will only take place if duplicated clones are removed.

Usage

```
processCGH(input, maxChromThreshold = 22, minChromThreshold = 1,
           method.of.averaging = NULL, ID = "ID", prop.missing = 0.1)
```

Arguments

input	Object of class MAList or RGList
maxChromThreshold	Chromosomes are ordered and numbered as usual, except for X and Y chromosome, which in for Homo sapiens genome have numbers 23 and 24 respectively, in for Mus musculus 20 and 21, etc. Remove chromosomes from segmentation analysis which are greater than this value.
minChromThreshold	Chromosomes are ordered and numbered as usual, except for X and Y chromosome, which in for Homo sapiens genome have numbers 23 and 24 respectively, in for Mus musculus 20 and 21, etc. Remove chromosomes from segmentation analysis which are lower than this value.

`method.of.averaging`
: If left as the default no combining of replicate spots takes place. Otherwise this should specify a function which takes a vector of duplicates and combines them into a single value.

`ID`
: Name of column in `RG$genes` corresponding to the clone names. For most data the default will work, however for affy data the value for ID should be `"CloneName"`

`prop.missing`
: For each probe the proportion of NA's is calculated, and the probe is kept for further analysis if the proportion of NA's is less than missing.prop

Value

Object of class `SegList`

Author(s)

Jane Fridlyand, Peter Dimitrov, John Marioni and Mike Smith

See Also

`MAList`

---

read.clonesinfo *Reading chromsome and positional information about each clone.*

Description

Function to read the chromosomal position information of each clone and incorperate it into the genes data.frame of the relevant object.

Usage

```
read.clonesinfo(file, RG, path = NULL, sep="\t", quote="\"")
```

Arguments

`file`
: Name of the file containing the chromosomal information.

`RG`
: Object containing a $genes data.frame that the information should be incorperated into.

`path`
: Path to the chromosomal information file.

`sep`
: Identifying the column seperator in the designated file.

`quote`
: Identifying the quotation character used in the designated file.

Author(s)

Mike Smith

readPositionalInfo *readPositionalInfo*

Description

This function automatically inserts information about the chromosomal postional of a clone into the $genes matrix of an RGList or MAList. This information is used in all the available segmentation methods as well as many of the plotting functions available in snapCGH.

Usage

```
readPositionalInfo(input, source, path = NULL)
```

Arguments

input	An object of class RGList or MAList
source	Defines which platform or technology this data was produced on. Currently supported options are: "aglient", "bluefuse", "nimblegen". This list will be expanded in time.
path	Optional parameter to specify where the original data is stored. Defaults to the current working directory. This option is only required for reading "bluefuse" data at the moment, as chromosome information isn't read by limma as default.

removeByWeights *Remove clones based on a weights matrix*

Description

An example function to be used by the filterClones method. This function takes an MA list, a weights matrix and a threshold and returns the indices of any clones with weight below the threshold.

Usage

```
removeByWeights(MA, weights=MA$weights, threshold = 0.2)
```

Arguments

MA	An object of class MAList
weights	A matrix with the same dimensions as MA containing weight information.
threshold	Threshold value. Any clones with weight below this are removed.

Author(s)

Mike Smith

See Also

filterClones

`runBioHMM` *This function implements the BioHMM*

Description

This function reads in a dataset of log2 ratios and the corresponding clone and covariate information. It calculates a heterogeneous HMM when there are 1,2,3,4 or 5 underlying states and chooses betwen them using either the AIC or BIC. It then assigns clones using a modified version of the Viterbi algorithm.

Usage

```
runBioHMM(input, useCloneDists = TRUE, covariates, criteria="AIC", delta=NA
,var.fixed=FALSE, epsilon = 1e-06, numiter = 30000)
```

Arguments

`input`	An object of class `MAList`or `SegList`
`useCloneDists`	Boolean stating whether the distance between clones should be incorportated into the HMM. If false then the HMM becomes homogeneous.
`covariates`	This is a dataframe containing information about covariate factors. The first two columns should be Chrom (giving the chromosome on which a clone is located) and Mb (giving the position of the chromosome along a particular chromosome in Megabases). The order should be the same as that described above with the following crucial difference. No covariate information about the first clone is used in the segmentation. Hence, for each chromosome, there should be one less row in the covariate dataframe than in the datainfo dataframe corresponding to this missing chromosome. This is important if the transition matrix is to be calculated correctly.
`criteria`	Options are AIC or BIC depending upon which we want to use to distinguish between the number of states
`delta`	A variable to be assigned if the BIC is used.
`var.fixed`	Logical variable - TRUE if you want to tie the variance to be the same across all states. Defaults to FALSE
`epsilon`	Stopping criterion for the optimization algorithm.
`numiter`	Number of iterations to be used in the optimization alogithm.

Value

The model returns an object of class `SegList`.

Author(s)

John Marioni and Mike Smith

References

Marioni, J.C., Thorne, N.P., Tavaré, S., BioHMM: a heterogeneous Hidden Markov Model for segmenting array CGH data, submitted

`runDNAcopy` *Results of segmenting an MAList data object using the DNAcopy library*

Description

The results of segmenting data from copy number array experiments from programs such as circular binary segmentation (CBS). This function requires the library `DNAcopy` to be loaded.

Usage

```
runDNAcopy(input, smooth.region=2, outlier.SD.scale = 4, smooth.SD.scale = 2, tr
    "perm"), kmax = 25, nmin = 200, undo.splits = c("none", "prune", "sdundo"),
    undo.prune = 0.05, undo.SD = 3, nperm = 10000, eta = 0.05)
```

Arguments

`input`	An object of class `MAList` or `SegList`
`smooth.region`	number of points to consider on the left and the right of a point to detect it as an outlier.
`outlier.SD.scale`	the number of SDs away from the nearest point in the smoothing region to call a point an outlier.
`smooth.SD.scale`	the number of SDs from the median in the smoothing region where a smoothed point is positioned.
`trim`	proportion of data to be trimmed for variance calculation for smoothing outliers and undoing splits based on SD.
`alpha`	significance levels for the test to accept change-points.
`p.method`	method used for p-value computation. For the "perm" method the p-value is based on full permutation. For the "hybrid" method the maximum over the entire region is split into maximum of max over small segments and max over the rest. Approximation is used for the larger segment max. Default is hybrid.
`kmax`	the maximum width of smaller segment for permutation in the hybrid method.
`nmin`	the minimum length of data for which the approximation of maximum statistic is used under the hybrid method.
`undo.splits`	A character string specifying how change-points are to be undone, if at all. Default is "none". Other choices are "prune", which uses a sum of squares criterion, and "sdundo", which undoes splits that are not at least this many SDs apart.
`undo.prune`	the proportional increase in sum of squares allowed when eliminating splits if undo.splits="prune".
`undo.SD`	the number of SDs between means to keep a split if undo.splits="sdundo".
`nperm`	number of permutations used for p-value computation.
`eta`	the probability to declare a change conditioned on the permuted statistic exceeding the observed statistic exactly j (= 1,...,nperm*alpha) times.

Value

The function returns an object of class SegList

Author(s)

Mike Smith, based upon DNAcopy help files written by E. S. Venkatraman and Adam Olshen

See Also

segment MAList runHomHMM runGLAD SegList

---

runGLAD *Results of segmenting an aCGHList data object using the GLAD library*

Description

This function allows the detection of breakpoints in genomic profiles obtained by array CGH technology and affects a status (gain, normal or lost) to each clone. It requires that the library GLAD is loaded.

Usage

```
runGLAD(input, smoothfunc="lawsglad", base=FALSE, sigma = NULL, bandwidth=10,
round=2, lambdabreak=8, lambdacluster=8, lambdaclusterGen=40,
type="tricubic", param=c(d=6), alpha=0.001, method="centroid",
nmax=8, verbose=FALSE, ...)
```

Arguments

input	An object of class MAList or SegList
smoothfunc	Type of algorithm used to smooth LogRatio by a piecewise constant function. Choose either lawsglad, aws or laws.
base	If TRUE, the position of clone is the physical position onto the chromosome, otherwise the rank position is used.
sigma	Value to be passed to either argument sigma2 of aws function or shape of laws. If NULL, sigma is calculated from the data.
bandwidth	Set the maximal bandwidth hmax in the aws or laws function. For example, if bandwidth=10 then the hmax value is set to 10*X_N where X_N is the position of the last clone.
round	The smoothing results are rounded or not depending on the round argument. The round value is passed to the argument digits of the round function.
lambdabreak	Penalty term (λ') used during the **Optimization of the number of breakpoints** step.
lambdacluster	Penalty term ($\lambda*$) used during the **MSHR clustering by chromosome** step.
lambdaclusterGen	Penalty term ($\lambda*$) used during the **HCSR clustering throughout the genome** step.

type	Type of kernel function used in the penalty term during the **Optimization of the number of breakpoints** step, the **MSHR clustering by chromosome** step and the **HCSR clustering throughout the genome** step.
param	Parameter of kernel used in the penalty term.
alpha	Parameter for ?????.
method	The agglomeration method to be used during the **MSHR clustering by chromosome** and the **HCSR clustering throughout the genome** clustering steps.
nmax	Maximum number of clusters (N*max) allowed during the the **MSHR clustering by chromosome** and the **HCSR clustering throughout the genome** clustering steps.
verbose	If TRUE some information are printed
...	

Details

For a detailed explanation of the GLAD algorithm please see the relevant section of the GLAD manual: glad

Value

The function returns an object of class SegList

See Also

glad MAList runHomHMM runDNAcopy SegList

---

runHomHMM *A function to fit unsupervised Hidden Markov model*

Description

This function fits an unsupervised Hidden Markov model to a given MAList or \code{MAList}

Usage

```
runHomHMM(input, vr = 0.01,
                maxiter = 100, criteria = "AIC", delta = NA,
                full.output = FALSE, eps = 0.01)
```

Arguments

input	an object of class MAList or SegList
vr	Gets passed to the function hidden as the pshape arguement.
maxiter	Gets passed to the function hidden as the iterlim arguement.
criteria	Choice of which selection criteria should be used in the algorithm. The choices are either AIC or BIC
delta	Delta value used of the BIC is selected. If no value is entered it defaults to 1.
full.output	if true the SegList output includes a probability that a clone is in its assigned state and a smoothed value for the clone.
eps	parameter controlling the convergence of the EM algorithm.

See Also

runDNAcopy runGLAD SegList

---

runTilingArray *Results of segmenting an MAList data object using the Picard et al algorithm found in the tilingArray library*

Description

Wrapper calling the Tiling Array segmentation algorithm on an MAList object. This function requires the library DNAcopy to be loaded.

Usage

```
runTilingArray(input, maxSeg = 5, maxk = 200, criteria = "BIC")
```

Arguments

input	An object of class MAList or SegList
maxSeg	integer of length 1, maximum number of segments (= 1 + maximum number of change points)
maxk	integer of length 1, maximum length of a single segment
criteria	Criteria for model selection. Options are "none", "AIC" and "BIC" (default)

Value

The function returns an object of class SegList

Author(s)

Mike Smith

See Also

segment MAList runHomHMM runGLAD SegList

SegList-class *Segmentation States - class*

Description

A list based class for storing the results of a segmentation algorithm. They are generally created by running one of the following functions runHomHMM, runGLAD or runDNAcopy on an MAList object.

Slots/List Components

Objects should contain the following list components:

pred:	Predicted value of the state.
disp:	Dispersion.
obs:	Observed value.
state:	Numeric value.
nstates.hmm:	The number of states per chromosome. Each row represents a chromosome and each column is an arra
genes:	data.frame that contains the chromosome and position on the chromosome for each clone. Used for plo

Optional:

rpred:	Smoothed value for the clone.
prob:	Probability of the clone being in the assigned state.

Methods

SegLists can be subsetted and combined. They also return dimensions so functions such as dim, nrow and ncol are also defined. SegList inherits the show method from the Limma class LargeDataObject. This means that the SegList will print in a relatively compact way.

Author(s)

Mike Smith

---

simulateData *A function for simulating aCGH data and the corresponding clone layout*

Description

This simulation scheme operates in two stages. Initially, we simulate the layout of clones before using a modified version of the scheme developed by Willenbrock et al., 2005 to generate the $\log_2$ ratios. For each simulated clone layout we generate 20 sets of simulated $\log_2$ ratios from one of five templates. Additionally, we also take account of the cellularity of the test sample in our simulation.

Usage

```
simulateData(nArrays = 20, chrominfo = NULL, prb.short.tiled = 0.5,
                 prb.long.tiled = 0.5, non.tiled.lower.res = 0.9,
                 non.tiled.upper.res = 1.1, length.clone.lower = 0.05,
                 length.clone.upper = 0.2, tiled.lower.res = -0.05,
                 tiled.upper.res = 0, sd = NULL, output = FALSE,
                 prb.proportion.tiled = c(0.2, 0.2, 0.2, 0.2, 0.2),
                 zerolengthnontiled = NULL, zerolengthtiled = NULL,
                 nonzerolengthnontiled = NULL, nonzerolengthtiled =
                 NULL, seed = 1)
```

Arguments

nArrays	The number of arrays we want to simulate
chrominfo	The information about chromosome length/centromere location to be used. Defaults to the information provided in aCGH package of Jane Fridlyand and Peter Dimitrov.
prb.short.tiled	The probability of a tiled region on the short arm of the simulated chromosome (defaults to 0.5).
prb.long.tiled	The probability of a tiled region on the long arm of the simulated chromosome (defaults to 0.5).
non.tiled.lower.res	The lower limit for the distance (in Mbs) between adjacent clones in non-tiled regions of the genome (defaults to 0.9Mb).
non.tiled.upper.res	The upper limit for the distance (in Mbs) between adjacent clones in non-tiled regions of the genome (defaults to 1.1Mb).
length.clone.lower	The lower limit for the length (in Mbs) of a clone (this defaults to 0.05Mb).
length.clone.upper	The upper limit for the length (in Mbs) of a clone (this defaults to 0.2Mb).
tiled.lower.res	The lower limit for the distance (in Mbs) between adjacent clones in tiled regions of the genome (defaults to -0.05Mb).
tiled.upper.res	The upper limit for the distance (in Mbs) between adjacent clones in tiled regions of the genome (defaults to 0Mb).
sd	The standard deviation of the simulated data in each of the states. Defaults to being randomly sampled between 0.1 and 0.2.
output	A logical variable which is TRUE if you want the output to be written to txt files in the present working directory. Defaults to FALSE.
prb.proportion.tiled	Given that an arm of a chromosome contains a tiled region this variable (which is a vector of length 5) gives the probability that 20,30,40,50 or 100% of the chromosome is tiled. It defaults to (0.2,0.2,0.2,0.2,0.2)
zerolengthnontiled	The empirical distribution for regions of the genome which are non-tiled and contain no copy number gains or losses. Defaults to zero.length.distr.non.tiled
zerolengthtiled	The empirical distribution for regions of the genome which are tiled and contain no copy number gains or losses. Defaults to zero.length.distr.tiled
nonzerolengthnontiled	The empirical distribution for regions of the genome which are non-tiled and contain no copy number gains or losses. Defaults to non.zero.length.distr.non.tiled
nonzerolengthtiled	The empiricial distribution for regions of the genome which are tiled and contain copy number gains or losses. Defaults to non.zero.length.distr.tiled
seed	Seed value allowing simulation to be reproduced if the same seed value is set.

Details

For more details see the article by Marioni and Thorne published in Bioinformatics.

Value

The function returns a list containing the following elements.

`clones`	Gives the start, end and midpoint of the simulated clones.
`class.output`	A list of the true underlying state clones are assigned to for each of the twenty simulations associated with each clone layout.
`class.matrix`	Defines the true underlying state clones are assigned to in each of the five classes
`classes`	Which of the five class outputs has been used to simulate the $\log_2$ ratios
`datamatrix`	A matrix containing twenty columns each of which contains the simulated $\log_2$ ratios associated with each of the simulations for a particular clone layout.
`samples`	Gives information about the cellularity associated with each of the samples.

Author(s)

John Marioni and Natalie Thorne

References

See the relevant article in Bioinformatics or the following website: www.damtp.cam.ac.uk/user/jcm68

`Viterbi.five` *A scaled Viterbi algorithm for allocating clones to one of five underlying states.*

Description

A work horse of the fit.model function. It uses a scaled version of the Viterbi algorithm to allocate clones to one of five underlying states as fitted using a heterogeneous HMM.

Usage

```
Viterbi.five(y, BFGS.output, BFGS.trans.mat)
```

Arguments

`y`	the data to be allocated to states
`BFGS.output`	The output obtained from the find.param.five function
`BFGS.trans.mat`	A list of the heterogeneous transition matrices

Value

A vector of numbers indicating to which state clones are allocated to.

Author(s)

John Marioni

`Viterbi.four` *A scaled Viterbi algorithm for allocating clones to one of four underlying states.*

Description

A work horse of the fit.model function. It uses a scaled version of the Viterbi algorithm to allocate clones to one of four underlying states as fitted using a heterogeneous HMM.

Usage

```
Viterbi.four(y, BFGS.output, BFGS.trans.mat)
```

Arguments

`y`	the data to be allocated to states
`BFGS.output`	The output obtained from the find.param.four function
`BFGS.trans.mat`	A list of the heterogeneous transition matrices

Value

A vector of numbers indicating to which state clones are allocated to.

Author(s)

John Marioni

`Viterbi.three` *A scaled Viterbi algorithm for allocating clones to one of two underlying states.*

Description

A work horse of the fit.model function. It uses a scaled version of the Viterbi algorithm to allocate clones to one of three underlying states as fitted using a heterogeneous HMM.

Usage

```
Viterbi.three(y, BFGS.output, BFGS.trans)
```

Arguments

`y`	the data to be allocated to states
`BFGS.output`	The output obtained from the find.param.three function
`BFGS.trans`	A list of the heterogeneous transition matrices

Value

A vector of numbers indicating to which state clones are allocated to.

Author(s)

John Marioni

`Viterbi.two` *A scaled Viterbi algorithm for allocating clones to one of two underlying states.*

Description

A work horse of the fit.model function. It uses a scaled version of the Viterbi algorithm to allocate clones to one of two underlying states as fitted using a heterogeneous HMM.

Usage

```
Viterbi.two(y, BFGS.output, BFGS.trans.mat)
```

Arguments

`y`	the data to be allocated to states
`BFGS.output`	The output obtained from the find.param.two function
`BFGS.trans.mat`	A list of the heterogeneous transition matrices

Value

A vector of numbers indicating to which state clones are allocated to.

Author(s)

John Marioni

`zero.length.distr.non.tiled` *Empirical distribution of segment lengths in non-tiled regions with no copy number gains or losses*

Description

This file contains the empirical distribution of segment lengths (of untiled regions and whose state mean indicates that they correspond to regions of no copy number gain or loss) derived by applying the DNAcopy segmentation scheme (Olshen et al., 2004) to an unpublished breast cancer dataset. Instead of using the physical length of the segments we calculate the lengths as a proportion of the length of the untiled region.

Usage

```
data(zero.length.distr.non.tiled)
```

Source

The empirical distribution was derived using an unpublished breast cancer dataset.

zero.length.distr.tiled

Empirical distribution of segment lengths in tiled regions with no copy number gains or losses

Description

This file contains the empirical distribution of segment lengths (of tiled regions and whose state mean indicates that they correspond to regions of no copy number gain or loss) derived by applying the DNAcopy segmentation scheme (Olshen et al., 2004) to an unpublished breast cancer dataset. Instead of using the physical length of the segments we calculate the lengths as a proportion of the length of the tiled region.

Usage

```
data(zero.length.distr.tiled)
```

Source

The empirical distribution was derived using an unpublished breast cancer dataset.

zoomChromosome *Interactive plot of a single chromsome*

Description

Plot splitting the screen into two. The top windows displays the entire chromosome, whilst the bottom plots a selected region. The plot is interactive allowing the user to click twice on a chromosome in the upper plot and have it the region between the two clicks displayed below.

Usage

```
zoomChromosome(..., array = 1, chrom.to.plot, colors = NULL, chrominfo =
chrominfo.Mb, ylim = c(-2, 2))
```

Arguments

`...`	Objects of type `MAList` or `SegList`
`array`	Specify which array should be plotted.
`chrom.to.plot`	Which chromosome should be plotted
`colors`	Vector specify the colors for each of the SegLists
`chrominfo`	chromosomal information associated with the mapping of the data.
`ylim`	Specify the limits of the y-axis

Details

If colors is unspecified then all SegLists passed to this function will be plotted in blue. Since this makes it quite hard to tell which is which it is highly recommened to specify the colors vector if more than one object is being passe to this function.

Author(s)

Mike Smith

`zoomGenome` *Interactive plot of the whole genome*

Description

Plot splitting the screen into two. The top windows displays the entire genome, whilst the bottom plots a single chromosome. The plot is interactive allowing the user to click on a chromosome in the upper plot and have it displayed below. Clicking to either side of the plot borders ends the interactivity.

Usage

```
zoomGenome(..., array = 1, colors = NULL, chrominfo = chrominfo.Mb)
```

Arguments

`...`	Objects of type `SegList`
`array`	Specify which array should be plotted.
`colors`	Vector specify the colors for each of the SegLists
`chrominfo`	chromosomal information associated with the mapping of the data.

Details

If colors is unspecified then all objects passed to this function will be plotted in blue. Since this makes it quite hard to tell which is which it is highly recommened to specify the colors vector if more than one object is being passe to this function.

Author(s)

Mike Smith

Index