Package ‘MPAgenomics’

February 19, 2015

Type Package
Title Multi-Patient Analysis of Genomic Markers
Version 1.1.2
Date 2014-10-29
Author Quentin Grimonprez with contributions from Guillemette Marot and Samuel Blanck. Some functions use code created by Sjoerd Vosse, Mark van de Wiel, Pierre Neuvial, Henrik Bengtsson.
Maintainer Samuel Blanck <samuel.blanck@inria.fr>
Description Preprocess and analysis of genomic data. MPAgenomics provides wrappers from commonly used packages to streamline their repeated manipulation, offering an easy-to-use pipeline. The segmentation of successive multiple profiles is performed with an automatic choice of parameters involved in the wrapped packages. Considering multiple profiles in the same time, MPAgenomics wraps efficient penalized regression methods to select relevant markers associated with a given outcome.
License GPL (>= 2)
Copyright Inria 2012-2014.
Imports R.utils, changepoint(>= 1.1), glmnet, cghseg, HDPenReg(>= 0.90), spikeslab
Suggests CGHcall, aroma.affymetrix, aroma.cn, aroma.core, aroma.light, snowfall, R.devices, R.filesets, R.methodsS3, R.oo, matrixStats
NeedsCompilation no
Repository CRAN
Date/Publication 2014-11-05 20:25:37

R topics documented:

MPAgenomics-package ......................................................... 2
addChipType ............................................................... 3
addData ................................................................. 3
callingObject ............................................................ 4
MPAgenomics-package

Multi-Patient Analysis of Genomic Markers

Description

This package provides functions to preprocess and analyze genomic data. The package was initially developed to select genomic markers associated with a given phenotype when several samples are available. In this context, markers refer to SNPs or copy number variations which are designed on the arrays.

The package also enables to preprocess all samples individually in order to keep maximum information from the original signals and improve the multi-patient analysis. In particular, this is useful to keep quantitative data for SNPs rather than usual genotype calls (AA, AB or BB) when these states are not relevant (e.g., in cancer studies where the number of copies differs from two copies).

Details

Package: MPAgenomics
Type: Package
Version: 1.1.2
Date: 2014-10-29
License: GPL (>=2)
addChipType

Author(s)
Quentin Grimonprez with contributions from Guillemette Marot and Samuel Blanck
Maintainer: Samuel Blanck <samuel.blanck@inria.fr>

Examples

# see the vignette for detailed examples
vignette("MPAgenomics")

---

addChipType  
*Add a new chip type to the existing aroma architecture*

Description
Create a folder in "annotationData/chipTypes" and copy the specified files in this folder.

Usage

addChipType(chipType, chipPath, verbose = TRUE)

Arguments
- chipType: Name of the new chip type to add.
- chipPath: Path to the files to add.
- verbose: Print additional information

Author(s)
Quentin Grimonprez

---

addData  
*Add a new data-set to the existing aroma architecture*

Description
Create a folder in "rawData" and copy the specified files in this folder.

Usage

addData(dataSetName, dataPath, chipType, verbose = TRUE)
Arguments

datasetName  Name of the data-set folder to create.
dataPath    Path of the folder containing the data CEL files.
chipType    Name of the used chip.
verbose    Print additionnal information.

Author(s)

Quentin Grimonprez

callingObject  

Create the list of parameters for callingProcess function

Description

create the list of parameters for callingProcess function

Usage

callingObject(copynumber, segmented, chromosome, position, featureNames, sampleNames)

Arguments

copynumber  A matrix containing the copy-number signal. Each column is a different patient.
segmented   A matrix containing the segmented copy-number signal. Matrix of the same size as copynumber.
chromosome  Chromosome associated with the copy-number signal.
position    Position of the signal.
featureNames Names of the probes (not necessary).
sampleNames Name of the sample (not necessary).

Value

a list in the right format for callingProcess function

Author(s)

Quentin Grimonprez
callingProcess

Calling aberrations in segmented copy-number signal.

Description
Launch the process of segmentation labeling. This function uses functions from CGHcall package developed by Sjoerd Vosse, Mark van de Wiel and Ilari Scheinin. See the CGHcall package for more details.

Usage
callingProcess(segmentData, nclass = 5, cellularity = 1, verbose = TRUE, ...)

Arguments
- segmentData: A list (see details).
- nclass: The number of levels to be used for calling. Either 3 (loss, normal, gain), 4 (including amplifications), 5 (including double deletions).
- cellularity: Proportion of tumor cells in the sample ranging from 0 to 1 (default=1). Reflects the contamination of the sample with healthy cells (1 = no contamination).
- verbose: If TRUE, print some details.
- ...: other options of CGHcall functions

Details
segmentData is a list containing:

- **copynumber**: A matrix. Each column contains a signal of copynumber for a profile. Each row corresponds to a genomic position of a probe.
- **segmented**: A matrix of the same size as copynumber. It contains the segmented signals.
- **chromosome**: A vector of length nrow(copynumber) containing the studied chromosome (number) for each position.
- **startPos**: A vector of length nrow(copynumber) containing the starting genomic position of each probe.
- **featureNames**: A vector of length nrow(copynumber) containing the names of each probe.
- **sampleNames**: A vector of length ncol(copynumber) containing the names of each profile.

Value
A list with the same element as segmentData list and

- **calls**: A matrix, of the same size as segmentData$copynumber matrix, containing the label of each point. -2=double loss, -1=loss, 0=normal, 1=gain, 2=amplification.
- **segment**: A data.frame that summarizes the different segments found.
probdlloss (if CGHcall was run with nclass=5) A matrix of the same size as segmentData$copynumber matrix. It contains the probability for each segmented copynumber to be a double loss.

probloss A matrix of the same size as segmentData$copynumber matrix. It contains the probability for each segment to be a loss.

probdnorm A matrix of the same size as segmentData$copynumber matrix. It contains the probability for each segment to be normal.

probdgain A matrix of the same size as segmentData$copynumber matrix. It contains the probability for each segment to be a gain.

probdamp (if CGHcall was run with nclass=4 or 5) A matrix of the same size as segmentData$copynumber matrix. It contains the probability for each segment to be an amplification.

Author(s)
Quentin Grimonprez

CNAobjectToCGHcallObject

Convert CNAobject

Description
convert CNA object (output of the function segment from DNAcopy package) into a list for the argument segmentData of the function callingProcess.

Usage
CNAobjectToCGHcallObject(CNAobject)

Arguments
CNAobject Output object of segment function from DNAcopy package

Value
a list at the required format of callingProcess.

Author(s)
Quentin Grimonprez

See Also
callingProcess
cnSegCallingProcess  

*Segment a copy-number signal and call the found segments.*

**Description**

This function applies the PELT method to segment each signal of the dataset and launches CGHcall for calling segments and detect aberrations. Results will be stored in a text file in the segmentation folder of the aroma architecture.

**Usage**

```r
cnSegCallingProcess(dataSetName, normalTumorArray, chromosome = 1:22,
                     method = c("cghseg", "PELT"), Rho = NULL, Kmax = 10,
                     listOffiles = NULL, onlySNP = TRUE, savePlot = TRUE, nclass = 3,
                     cellularity = 1, ...)
```

**Arguments**

- **dataSetName**: name of the data-set folder in the rawData folder containing the signals to use.
- **normalTumorArray**: Only in the case of normal-tumor study. A csv file or a data.frame containing the mapping between normal and tumor files. The first column contains the name of normal files and the second the names of associated tumor files.
- **chromosome**: A vector containing the chromosomes to segment.
- **method**: method of segmentation, either "PELT" or "cghseg".
- **Rho**: For method="PELT", vector containing all the penalization values to test for the segmentation. If no values are provided, default values will be used.
- **Kmax**: For method="cghseg", maximal number of segments.
- **listOffiles**: A vector containing the names of the files from the dataSetName to use.
- **onlySNP**: If TRUE, only the SNP probes will be used.
- **savePlot**: If TRUE, save the segmented signal in figures folder.
- **nclass**: The number of levels to be used for calling. Either 3 (loss, normal, gain), 4 (including amplifications), 5 (including double deletions) (default=3).
- **cellularity**: Percentage of tumored cells in the sample (default=1).
- **...**: Other parameters of CGHcall function

**Value**

A data.frame containing columns:

- **sampleNames**: Name of the file.
- **chrom**: The chromosome of the segment.
- **chromStart**: The starting position (in bp) of a segment. This position is not included in the segment.
createArchitecture

**Description**

Create the architecture required by aroma.* packages and copy files into created folders.

**Usage**

```r
createArchitecture(dataSetName, chipType, dataSetPath, chipFilesPath,
                   path = ".", verbose = FALSE, tags = NULL)
```

**Arguments**

- **dataSetName** The name of the data-set folder to create.
- **chipType** The name of the used chip.
- **dataSetPath** Path to the folder containing the data CEL files.
- **chipFilesPath** Path to the folder containing the chip files.
- **path** Path where the architecture should be created (default= ".").
- **verbose** Print information during the process (default=FALSE).
- **tags** Common tag which appears in the different file names (cdf, ugp, ufl) of the chip. For no tag, use tags=NULL (default = NULL). See details for more information.

**Details**

All the cdf chip file names must follow the following rule: `<chipType>_<Tags>.cdf`

Multiple tags must be separated by a comma. If there is no tag, the pattern is `<chipType>.cdf`

**Author(s)**

Quentin Grimonprez
createEmptyArchitecture

See Also
copyChipFiles, copyDataFiles, createAromaArchitecture

Examples

#DO NOT EXECUTE before reading of the vignette
#createArchitecture("test1","GenomeWideSNP_6","./celPATH","./chipPATH",path=".",TRUE,"Full")

createEmptyArchitecture

Create aroma architecture

Description

Create the architecture required by aroma packages

Usage

createEmptyArchitecture(dataSetName, chipType, path = ".", verbose = TRUE)

Arguments

dataSetName name of the data set
chipType type of the chip used for obtaining the data
path path where folders are created
verbose if TRUE, print details of the process

Details

This function creates the following architecture: Architecture to create: <path> +- annotationData/ | +- chipTypes/ | +- <chipType>/ <- must match exactly the name of the CDF file (fullname minus tags) | +- CDF file(s) and other annotation (possibly subdirectories) | +- rawData/ +- <dataSetName>/ +- <chipType>/ <- must match exactly a chip type folder under annotationData/ +- CEL files

Author(s)

Quentin Grimonprez
filterSeg

Filter segments

Description

This function filters the output of a segmentation and label process. It allows to keep only segments over a minimal length or containing at least a minimal number of probes.

Usage

```r
filterSeg(segmentList, minLength = 1L, minProbes = 1L, keptLabel = c("loss", "gain"))
```

Arguments

- `segmentList`: A data.frame containing a description of segments, it must have at least columns named "chromStart", "chromEnd", "probes" and "calls". (see the output of `cnSegCallingProcess` function).
- `minLength`: The minimum length (in bp) for a segment. All the shorter segments are removed.
- `minProbes`: The minimum number of probes for a segment. All the segments with less probes are removed.
- `keptLabel`: Vector of labels to keep. Only segment with one of the specified label will be kept.

Value

A data.frame of the same format as `segmentList`.

Author(s)

Quentin Grimonprez

findPlateau

Find the best choice of segmentation parameter.

Description

From the results of a segmentation of a signal for different values of a segmentation parameter rho, this function will search an optimal value of rho corresponding to the biggest plateau (stabilization in the number of breakpoints).

Usage

```r
findPlateau(resSeg, Rho, plot = TRUE, verbose = TRUE)
```
getCopyNumberSignal

Arguments

resSeg a list, each element of the list is a vector with the breakpoints for a value of Rho.
Rho vector with the values of Rho.
plot if TRUE, some graphics will be plotted.
verbose if TRUE print some informations.

Value

a list containing:

rho Optimal parameter found.
maxPlateau A vector with the first and the last position of the biggest plateau.
plateau A matrix of 3 columns, each row corresponds to a different plateau. The first column is the
starting value of a plateau, the second, the length of the plateau and the third, the number of
values of rho contained in the plateau.

Author(s)

Quentin Grimonprez

getchCopyNumberSignal

Extract copy-number signal from aroma files

Description

Extract copy-number signals from aroma files. It requires to have executed the normalization pro-
cess suggested by aroma packages, by using signalPreProcess for example.

Usage

getCopyNumberSignal(datasetName, chromosome, normalTumorArray,
onlySnp = FALSE, listoffiles = NULL, verbose = TRUE)

Arguments

dataSetName The name of the data-set folder (it must correspond to a folder name in rawData
directory).
chromosome A vector containing the chromosomes for which the signal will be extracted.
normalTumorArray Only in the case of normal-tumor study. A csv file or a data.frame containing the
mapping between normal and tumor files. The first column contains the name
of normal files and the second the names of associated tumor files.
onlySnp If TRUE, only the copy-number for SNPs positions will be returned (default=FALSE).
listoffiles A vector containing the names of the files in dataSetName folder for which the
copy-number profiles will be extracted (default is all the files).
verbose If TRUE print some information (default=TRUE).
getFracBSignal

Details

The aroma architecture must be respected. The working directory must contain rawData folder and totalAndFracBData folder. To easily access the names of the files available in a dataset, one can use the getListOfFiles function.

Value

a list of length the number of chromosomes containing a data.frame with columns:

- **chromosome** Chromosome of the signal.
- **position** Positions associated with the copy-number.
- **copynumber** Copy number profiles of selected files; the name of each column is the name of the associated data file name.
- **featureNames** Names of the probes.

Author(s)

Quentin Grimonprez

Examples

#DO NOT EXECUTE before reading the vignette
C=c=getCopyNumberSignal("data1",5,normalTumorArray,TRUE)
C=c=getCopyNumberSignal("data2",5,onlySNP=TRUE)

------

getFracBSignal  *Extract allele B fraction signal from aroma files*

Description

Extract allele B fraction signals from aroma files. It requires to have executed the normalization process suggested by aroma packages, by using signalPreProcess for example.

Usage

getFracBSignal(dataSetName, chromosome, normalTumorArray, listOffiles = NULL, 
verbose = TRUE)

Arguments

dataSetName  The name of the data-set folder (it must correspond to a folder name in rawData folder.)

chromosome  A vector containing the chromosomes for which the allele B fraction signal must be extract.
getGenotypeCalls

normalTumorArray
 Only in the case of normal-tumor study. A csv file or a data.frame containing
the mapping between normal and tumor files The first column contains the name
of normal files and the second the names of associated tumor files.

listOfFiles
 A vector containing the names of the files in dataSetName folder for which the
allele B fraction profiles will be extracted (default is all the files).

verbose
 If TRUE print some information (default=TRUE).

Details

The aroma architecture must be respected. The working directory must contain rawData folder and
totalAndFracBData folder. To easily access the names of the files available in a dataset, one can use
the getListOfFiles function.

Value

a list of length the number of chromosomes containing a list of two elements (normal and tumor)
containing a data.frame with columns:

 chromosome Chromosome of the signal.
 position Positions associated with the allele B fraction.
 fracB Allele B fraction profiles of selected files; the name of each column is the name of the
 associated data file name.
 featureNames Names of the probes.

Author(s)

Quentin Grimonprez

Examples

#DO NOT EXECUTE before reading the vignette
#fracB=getFracBSignal("data1",5,normalTumorArray)
#fracB=getFracBSignal("data2",5)

getGenotypeCalls Extract genotype calls from aroma files

Description

Extract genotype calls from aroma files. It requires to have executed the normalization process
suggested by aroma packages, by using signalPreProcess for example.

Usage

getGenotypeCalls(dataSetName, chromosome, listOfFiles = NULL,
   verbose = TRUE)
getListOfFiles

Arguments

- **dataSetName** The name of the data-set folder (it must correspond to a folder name in rawData folder.)
- **chromosome** A vector containing the chromosomes for which the genotype call will be extracted.
- **listOfFile** A vector containing the names of the files in dataSetName folder for which the genotype signal will be extracted (default is all the files).
- **verbose** If TRUE print some information (default=TRUE)

Details

The aroma architecture must be respected. The working directory must contain rawData folder and totalAndFracBData folder. To easily access the names of the files available in a dataset, one can use the `getListOfFiles` function.

Value

A list of length the number of chromosomes containing a data.frame with columns:

- **chromosome** Chromosome of the signal.
- **position** Positions associated with the genotype.
- **genotype** Genotype calls corresponding to selected files; the name of each column is the name of the associated data file name.
- **featureNames** Names of the probes.

Author(s)

Quentin Grimonprez

Examples

```r
#DO NOT EXECUTE before reading the vignette
fracB=getGenotypeCalls("data1",5)
```

---

**getListOfFiles**

Get the contents of a data folder

Description

Get the cel files of the specified dataSetName

Usage

`getListOfFiles(dataSetName, chipType)`
getSymFracBSignal

Arguments

- **dataSetName** The name of a data-set folder
- **chipType** The name of the used chip

Details

If chipType is not provided, the function returns the files for the first chip (in the alphabetic order).

Value

The filenames of all the files in rawData/dataSetName/chipType

Author(s)

Quentin Grimonprez

getsymfracbsignal  Extract symmetrized allele B fraction signal from aroma files

Description

Extract symmetrized allele B fraction signals from aroma files. It requires to have executed the normalization process suggested by aroma packages, by using `signalPreProcess` for example.

Usage

```r
getSymFracBSignal(dataSetName, file, chromosome, normalTumorArray, verbose = TRUE)
```

Arguments

- **dataSetName** The name of the data-set folder (it must correspond to a folder name in rawData folder.)
- **file** The name of the file in dataSetName to extract.
- **chromosome** A vector with the chromosomes for which the symetrized signal will be extracted.
- **normalTumorArray** Only in the case of normal-tumor study. A csv file or a data.frame containing the mapping between normal and tumor files. The first column contains the name of normal files and the second the names of associated tumor files.
- **verbose** If TRUE, print some informations.

Details

The aroma architecture must be respected. The working directory must contain rawData folder and totalAndFracBD ata folder. To easily access the names of the files available in a dataset, one can use the `getListOfFiles` function.
**Value**

a list of length the number of chromosome containing a data.frame with columns:

- **chromosome** chromosome corresponding to the signal.
- **position** Positions associated to the allele B fraction.
- **fracB** One column named by the data file name. It contains the symmetrized allele B fraction signal for the specified profile.
- **featureNames** Names of the probes.

**Author(s)**

Quentin Grimonprez

**Examples**

```R
#DO NOT EXECUTE
fracB=getAsymfracbsignal("data1",5,normalTumorArray)
fracB=getAsymfracbsignal("data2",5)
```

---

**Description**

This function transforms the two matrices CN and fracB in one matrix which is used in the lars algorithm. Each signal is weighted

**Usage**

```R
HDLarsbivariate(CN, fracB, y, weightsCN = 1/apply(CN, 1, sd),
    weightsfracB = 1/apply(fracB, 1, sd), meanCN = 2, maxSteps, eps)
```

**Arguments**

- **CN** matrix containing copy-number signals. Each row corresponds to a different signal.
- **fracB** matrix containing copy-number signals. Each row corresponds to a different signal.
- **y** vector containing the response associated to each signal
- **weightsCN** vector of length nrow(CN); weights associated to each signal for the copy-number signal
- **weightsfracB** vector of length nrow(fracB); weights associated to each signal for the copy-number signal
- **meanCN** value for centering the copy-number signal (default value = 2)
- **maxSteps** maximum number of steps for the lars algorithm
- **eps** tolerance
Value

a LarsPath object

Author(s)

Quentin Grimonprez

markerSelection

markerSelection

Description

This function selects, for each chromosome, the most relevant markers according to a response.

Usage

markerSelection(dataSetName, dataResponse, chromosome = 1:22,
signal = c("CN", "fracB"), normalTumorArray, onlySNP = FALSE,
nbFolds = 10, loss = c("logistic", "linear"), plot = TRUE,
 pkg = c("HDPenReg", "spikeslab"), ...)

Arguments

dataSetName The name of the data-set folder.
dataResponse A csv files or a data.frame with 2 columns : "files" and "response". The column "files" contains the filename to extract and the second column the response associated to the file.
chromosome A vector containing the number of the chromosomes for the SNPs selection.
signal either "CN" or "fracB". corresponding to which signal will be analyzed (default="CN").
normalTumorArray Only in the case of normal-tumor study. A csv file or a data.frame containing the mapping between normal and tumor files. The first column contains the name of normal files and the second the names of associated tumor files.
onlySNP (only if signal="CN"). If TRUE, only the SNPs probes are used (default=FALSE).
nbFolds number of folds in the cross validation (default=10).
loss either "logistic" (binary response) or "linear" (quantitative response), default is "logistic"
plot If TRUE, cross-validation mean squared error is plotted (default=TRUE).
pkg Either "HDPenReg" or "spikeslab". Ued package in linear case.
... Other parameters for HDlars, glmnet or spikeslab function.
Details

This function requires to use the aroma folder architecture. In your working directory, there must have the rawData folder and totalAndFracBData folder. This function launches the lars algorithm on the CN or fracB data and uses a cross-validation to select the most appropriate solution.

Value

a list containing length(chromosome) elements. Each element is a list containing

- chr chromosome corresponding to the signal.
- markers.index A vector containing the index of all selected markers.
- markers.position A vector containing the position of all selected markers.
- markers.names A vector containing the names of all selected markers.
- coefficient A vector containing the coefficients of all selected markers.
- intercept Intercept of the model.

Author(s)

Quentin Grimonprez

See Also

HDPenReg, glmnet, spikeslab

segFracBSignal  segmentation function for the allele B fraction

Description

This function launches the segmentation of allele B fraction only for heterozygous SNPs.

Usage

segFracBSignal(dataSetName, normalTumorArray, chromosome = 1:22,
    method = c("cghseg", "PELT"), Rho = NULL, Kmax = 10,
    listOffiles = NULL, savePlot = TRUE, verbose = TRUE)

Arguments

dataSetName The name of the data-set folder (it must correspond to a folder name in rawData folder).

normalTumorArray Only in the case of normal-tumor study. A csv file or a data.frame containing the mapping between normal and tumor files. The first column contains the name of normal files and the second the names of associated tumor files.

chromosome A vector with the chromosomes to be segmented.
method: method of segmentation, either "PELT" or "cghseg".
Rho: For method="PELT", vector containing all the penalization values to test for the segmentation. If no values are provided, default values will be used.
Kmax: For method="cghseg", maximal number of segments.
listOfFiles: A vector containing the names of the files in dataSetName folder for which the allele B profile is segmented (default is all the files).
savePlot: if TRUE, graphics of the segmented allele B profile will be saved in the figures/dataSetName/segmentation/fracB folder. (default=TRUE).
verbose: if TRUE print some informations

Value

a data.frame where each row correspond to a different segment with columns:
sampleNames: The name of the signal.
chromosome: A vector of the same size as copynumber containing the chromosome number.
chromStart: The starting position of a segment.
chromEnd: The ending position of a segment.
probes: The number of probes in the segment.
means: Means of the segment.

Author(s)

Quentin Grimonprez

segmentation function

Description

This function launches the segmentation of a signal.

Usage

segmentation(signal, method = c("cghseg", "PELT"), Rho = NULL, Kmax = 10,
position = NULL, plot = TRUE, verbose = TRUE)

Arguments

signal: a vector containing the signal.
method: method of segmentation, either "PELT" or "cghseg".
Rho: For method="PELT", vector containing all the penalization values to test for the segmentation. If no values are provided, default values will be used.
Kmax: For method="cghseg", maximal number of segments.
position: A vector containing the position of all elements of the signal (not necessary)
plot: if TRUE, plot the segmentation results
verbose: if TRUE print some informations
segmentationAroma

Value

- **signal** A vector containing the signal.
- **segmented** A vector of the same size as signal containing the segmented values.
- **startPos** The position of each probe.
- **segment** A data.frame that summarizes the results of the segmentation. Each row is a different segment with the start position, end position, number of points in the signal and the value of the segment.

Author(s)

Quentin Grimonprez

---

**segmentationAroma**  
**segmentation function**

Description

This function launches the segmentation process using the aroma architecture.

Usage

```r
segmentationAroma(dataSetName, normalTumorArray, chromosome = 1:22,  
method = c("cghseg", "PELT"), kmax = NULL, rho = NULL, listOffiles = NULL,  
onlySNP = TRUE, savePlot = TRUE, verbose = TRUE)
```

Arguments

- **dataSetName** The name of the data-set folder (it must correspond to a folder name in rawData folder).
- **normalTumorArray** Only in the case of normal-tumor study. A csv file or a data.frame containing the mapping between normal and tumor files The first column contains the name of normal files and the second the names of associated tumor files.
- **chromosome** A vector with the chromosomes to be segmented.
- **method** method of segmentation, either "PELT" or "cghseg".
- **kmax** For method="cghseg", maximal number of segments.
- **rho** For method="PELT", vector containing all the penalization values to test for the segmentation. If no values are provided, default values will be used.
- **listOffiles** A vector containing the names of the files in dataSetName folder for which the copy number profiles will be segmented (default is all the files).
- **onlySNP** If TRUE, only the copy-number for SNPs positions will be returned (default=TRUE).
- **savePlot** if TRUE, graphics of the segmented CN signal will be saved in the figures/dataSetName/segmentation/CN folder. (default=TRUE).
- **verbose** if TRUE print some informations
Value

a list containing

- `copynumber` A vector containing the copynumber signal.
- `segmented` A vector of the same size as `copynumber` containing the segmented values.
- `startPos` The position of each probes.
- `chromosome` A vector of the same size as `copynumber` containing the chromosome number.
- `featureNames` Names of the probes.
- `sampleNames` The name of the signal.

`segment` A data.frame that summarizes the results of the segmentation. Each row is a different segment with the chromosome, start position, end position, number of probes in the signal and the value of the segment.

Author(s)

Quentin Grimonprez

---

**segmentationObject**  
*Create the list of parameters for segmentation function*

Description

create the list of parameters for segmentation function

Usage

`segmentationObject(copynumber, chromosome, position, featureNames, sampleNames)`

Arguments

- `copynumber` A vector containing the copy-number signal for one patient and one chromosome.
- `chromosome` Chromosome associated with the copy-number signal.
- `position` Position of the signal.
- `featureNames` Names of the probes (not necessary).
- `sampleNames` Name of the sample (not necessary).

Value

a list in the right format for segmentation function

Author(s)

Quentin Grimonprez
SignalNormalization  Normalization process

Description

low-level normalization process for estimating raw copy-numbers and allele B fraction.

Usage

SignalNormalization(dataFolder, chipType, normalTumorArray,
  genotypeCallsMethod = "naive", savePlot = TRUE, tags = NULL)

Arguments

dataFolder     Name of the data set.
chipType       Type of the chip used for the data.
normalTumorArray
  Only in the case of normal-tumor study. A csv file or a data.frame containing the
  mapping between normal and tumor files. The first column contains the name
  of normal files and the second the names of associated tumor files.
genotypeCallsMethod
  method used for genotypage, default is "naive".
savePlot       If TRUE, graphics of the CN signal and allele B fraction signal will be saved in
  the figures folder.
tags           Common tag which appears in the different file names (cdf, ugp, ufl) of the chip.
  For no tag, use tags=NULL (default = NULL). See details for more information.

Details

The aroma architecture must be respected: <working directory> +- annotationData/ | +- chipTypes/
  | +- <chipType>/ <- must match exactly the name of the CDF file (fullname minus tags) | +- CDF
  file(s) and other annotation (possibly subdirectories) | +- rawData/ +- <nameOfDataSet>/ +- <chip-
  Type>/ <- must match exactly a chip type folder under annotationData/ +- CEL files

All the cdf chip file names must follow the following rule: <chipType>,<Tags>.cdf

Multiples tags must be separated by a comma. If there is no tag, the pattern is <chipType>.cdf

Author(s)

Quentin Grimonprez
signalPreProcess

**Normalization process**

**Description**

Normalization process for estimating raw copy-numbers and allele B fraction.

**Usage**

```r
signalPreProcess(dataSetName, chipType, normalTumorArray, dataSetPath,
chipFilesPath = dataSetPath, path = ".", createArchitecture = TRUE,
savePlot = TRUE, tags = NULL)
```

**Arguments**

- `dataSetName` Name of the data set. If you use `architecture=FALSE`, the name must correspond to a name of folder in the `rawData` folder.
- `chipType` Type of the used chip (e.g. "GenomeWideSNP_6"). If `architecture=FALSE`, the files of the chip must be contained in the `annotationData` folder, if `TRUE`, they have to be in the "chipTypePath" folder.
- `normalTumorArray` Only in the case of normal-tumor study. A csv file or a data.frame containing the mapping between normal and tumor files. The first column contains the name of normal files and the second the names of associated tumor files.
- `dataSetPath` (only if `createArchitecture=TRUE`) Path to the folder containing the CEL files of the data-set.
- `chipFilesPath` (only if `createArchitecture=TRUE`) Path to the folder containing all the annotations files for the specified chip type.
- `path` (only if `createArchitecture=TRUE`) Path where the architecture should be created (default=".").
- `createArchitecture` if `TRUE`, the aroma architecture will be automatically created (default=TRUE). CEL files of the data and chip files will be copied (not moved).
- `savePlot` if `TRUE`, graphics of the CN signal and allele B fraction signal will be saved in the figures/signal folder.
- `tags` Common tag which appears in the different file names (cdf, ugp, ufl) of the chip. For no tag, use `tags=NULL` (default = NULL). See details for more information.

**Details**

The following architecture must be used: `<working directory> |- annotationData/ |- chipTypes/ |- <chipType>/ <- must match exactly the name of the CDF file (fullname minus tags) |- CDF file(s) and other annotation (possibly subdirectories) |- rawData/ |- <nameOfDataSet>/ +- <chipType>/ <- must match exactly a chip type folder under annotationData/ +- CEL files`
If you use `createArchitecture=TRUE`, this function creates this architecture for you and copy your files in the right folders.

The functions will create other folders which contain figures, results of normalization.

If you already have the required architecture, you just have to add your data in the `rawData` folder with respect to the architecture.

All the cdf chip file names must follow the following rule : `<chipType>,<Tags>.cdf`

Multiples tags must be separated by a comma. If there is no tag, the pattern is `<chipType>.cdf`

**Author(s)**

Quentin Grimonprez

---

```r
symmetrizeFracB

**symmetrize an allele B fraction signal**

**Description**

The allele B fraction signal is the ratio between the signal from the allele B and the total signal. The symmetrization of the fraction allele B signal x is : \(2 \times \text{abs}(x - 0.5)\).

**Usage**

```r
symmetrizeFracB(fracB)
```

**Arguments**

- `fracB` a vector containing an allele B fraction signal.

**Value**

a vector containing the symmetrized signal.

**Author(s)**

Quentin Grimonprez

**Examples**

```r
signalA=abs(rnorm(100))
signalB=abs(rnorm(100))
signalFracB=signalA/(signalA+signalB)
symFracB=symmetrizeFracB(signalFracB)
```
variableSelection

SNPs selection

Description
This function selects the most relevant variables according to a response.

Usage
variableSelection(dataMatrix, dataResponse,
    nbFolds = min(length(dataResponse), 10), loss = c("logistic", "linear"),
    plot = TRUE, pkg = c("HDPenReg", "spikeslab"), ...)

Arguments

dataMatrix       Matrix containing the data, each row is a different sample.
dataResponse     response associated to the data.
nbFolds          number of folds in the cross validation.
loss              either "logistic" (binary response) or "linear" (quantitative response).
plot             If TRUE plot cross-validation mean squared error (default=TRUE).
pkg              Either "HDPenReg" or "spikeslab". Used package in linear case.
...              supplementary arguments for cv.glmnet function in case of logistic loss or for HDlars or spikeslab function for linear loss.

Value

a list containing

variable A vector containing the index of all selected variables.
coefficient A vector containing the coefficients of all selected variables.
intercept Intercept of the model.

Author(s)
Quentin Grimonprez
Index

+Topic package
  MPagenomics-package, 2

addChipType, 3
addData, 3

callingObject, 4
callingProcess, 4, 5, 6
CNAobjectToCGHcallObject, 6
cnSegCallingProcess, 7, 10
createArchitecture, 8
createEmptyArchitecture, 9

filterSeg, 10
findPlateau, 10

getchCopyNumberSignal, 11
getFracBSignal, 12
getGenotypeCalls, 13
getListOffiles, 12–14, 14, 15
getSymFracBSignal, 15

HDlarsbivariate, 16

markerSelection, 17
MPagenomics (MPagenomics-package), 2
MPagenomics-package, 2
MPagenomics-package,
   (MPagenomics-package), 2

segFracBSignal, 18
segmentation, 19, 21
segmentationAroma, 20
segmentationObject, 21
SignalNormalization, 22
signalPreProcess, 11–13, 15, 23
symmetrizeFracB, 24

variableSelection, 25