CGHcall

April 19, 2009

`CGHcall-package` *Calling aberrations for array CGH tumor profiles.*

Description

Calls aberrations for array CGH data using a six state mixture model as well as several biological concepts that are ignored by existing algorithms. Visualization of profiles is also provided.

Details

Package:	CGHcall
Type:	Package
Version:	2.0.0
Date:	2007-03-14
License:	GPL

Author(s)

Sjoerd Vosse and Mark van de Wiel

Maintainer: Sjoerd Vosse <s.vosse@vumc.nl>

References

Mark A. van de Wiel, Kyung In Kim, Sjoerd J. Vosse, Wessel N. van Wieringen, Saskia M. Wilting and Bauke Ylstra. CGHcall: calling aberrations for array CGH tumor profiles. *Bioinformatics, 23*, 892-894.

`CGHcall` *Calling aberrations for array CGH tumor profiles.*

Description

Calls aberrations for array CGH data using a six state mixture model.

Usage

```
CGHcall(inputSegmented, prior = "auto", nclass = 3, organism = "human")
```

Arguments

`inputSegmented`	An object of class `cghSeg`
`prior`	Options are `all`, `not all`, or `auto`. See details for more information.
`nclass`	The number of levels to be used for calling. Either `3` (loss, normal, gain) or `4` (including amplifications).
`organism`	Either `human` or `other`. This is only used for chromosome arm information when `prior` is set to `all` or `auto` (and samplesize > 20).

Details

Please read the article and the supplementary information for detailed information on the algorithm.

The parameter `prior` states how the data is used to determine the prior probabilities. When set to `all`, the probabilities are determined using the entire genome of each sample. When set to `not all` probabilites are determined per chromosome for each sample when `organism` is set to `other` or per chromosome arm when `organism` is `human`. The chromosome arm information is taken from the March 2006 version of the UCSC database. When `prior` is set to `auto`, the way probabilities are determined depends on the sample size. The entire genome is used when the sample size is smaller than 20, otherwise chromosome (arm) information is used.

Please note that CGHcall uses information from all input data to determine the aberration probabilities. When for example triploid or tetraploid tumors are observed, we advise to run CGHcall separately on those (groups of) samples.

Value

This function return a list with three components:

`probabilities`	A dataframe with 3 columns of probe information (name, chromosome and position), followed by k columns with aberration probabilities for each sample, where k is the number of levels used for calling (`nclass`).
`calls`	A dataframe with the calls for each sample. Values are `-1` (loss), `0` (normal) or `1` (gain). If 4 levels were used for calling, a value of `2` represents an amplification.
`segments`	A matrix with the segments for each profile. The first column contains the sample number. The second column the level of the current segment and the third and fourth columns the start and end of the segment in probe number respectively.

Author(s)

Sjoerd Vosse & Mark van de Wiel

References

Mark A. van de Wiel, Kyung In Kim, Sjoerd J. Vosse, Wessel N. van Wieringen, Saskia M. Wilting and Bauke Ylstra. CGHcall: calling aberrations for array CGH tumor profiles. *Bioinformatics, 23*, 892-894.

Examples

```
data(WiltingSeg)
## Not run: result <- CGHcall(WiltingSeg)
```

`normalize` *Normalization and cellularity adjustment for arrayCGH data.*

Description

This function normalizes arrayCGH data using the global mode or median. It can also adjust for the cellularity of your data.

Usage

```
normalize(input, method = "median", cellularity = 1, smoothOutliers = TRUE, ...)
```

Arguments

`input`	Object of class `cghRaw`.
`method`	Normalization method, either 'median', 'mode', or 'none'.
`cellularity`	A vector of cellularities ranging from 0 to 1 to define the contamination of your sample with healthy cells (1 = no contamination). See details for more information.
`smoothOutliers`	Logical. Indicates whether outliers should be smoothed using the `smooth.CNA` function.
`...`	Arguments for `smooth.CNA`.

Details

The cellularity parameter should be a vector of length n where n is the number of samples in your dataset. The vector is recycled if there are not enough values in it, or truncated if there are too many. For more information on the correction we refer to section 1.6 of the supplementary information for van de Wiel et al. 2006.

Value

This function returns a dataframe in the same format as the input with normalized and/or cellularity adjusted log2 ratios.

Author(s)

Sjoerd Vosse & Mark van de Wiel

Examples

```
data(WiltingData)
## Convert to \code{\LinkA{cghRaw}{cghRaw}} object
cgh <- cghRaw(WiltingData)
## First preprocess the data
raw.data <- preprocess(cgh)
## Simple global median normalization for samples with 75% tumor cells
perc.tumor <- rep(0.75, 3)
normalized.data <- normalize(raw.data, cellularity=perc.tumor)
```

`preprocess` *Preprocess arrayCGH data*

Description

This function preprocesses your aCGH data so it can be processed by other functions without errors.

Usage

```
preprocess(input, maxmiss = 30, nchrom = 22, ...)
```

Arguments

`input`	Object of class `cghRaw`.
`maxmiss`	Maximum percentage of missing values per row.
`nchrom`	Number of chromosomes.
`...`	Arguments for `impute.knn` from the impute package.

Details

This function performs the following actions on arrayCGH data:

- Filter out data with missing position information.
- Remove data on chromosomes larger than nchrom.
- Remove rows with more than maxmiss percentage missing values.
- Imputes missing values using the `impute.knn` function from the impute package.

Value

This function returns a dataframe in the same format as the input with missing values imputed.

Author(s)

Sjoerd Vosse & Mark van de Wiel

References

Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B. Altman (2001). Missing value estimation methods for DNA microarrays. *Bioinformatics, 17*, 520-525.

Examples

```
data(WiltingRaw)
preprocessed <- preprocess(WiltingRaw)
```

`segmentData` *Breakpoint detection for arrayCGH data.*

Description

A wrapper function to run existing breakpoint detection algorithms on arrayCGH data. Currently only DNAcopy is implemented.

Usage

```
segmentData(input, method = "DNAcopy", ...)
```

Arguments

`input`	Object of class `cghRaw`.
`method`	The method to be used for breakpoint detection. Currently only 'DNAcopy' is supported, which will run the `segment` function.
`...`	Arguments for `segment`.

Details

See `segment` for details on the algorithm.

Value

This function returns a dataframe in the same format as the input with segmented arrayCGH data.

Author(s)

Sjoerd Vosse & Mark van de Wiel

References

Venkatraman, A.S., Olshen, A.B. (2007). A faster circulary binary segmentation algorithm for the analysis of array CGH data. *Bioinformatics, 23*, 657-663.

Examples

```
data(WiltingNorm)
## Not run: segmented.data <- segmentData(WiltingNorm, alpha=0.02)
```

Index