Package: ARRmNormalization (via r-universe)

October 10, 2026

Type Package

Title Adaptive Robust Regression normalization for Illumina methylation data

Version 1.53.0

Date 2013-01-10

Author Jean-Philippe Fortin, Celia M.T. Greenwood, Aurelie Labbe.

Depends R (>= 2.15.1), ARRmData

Maintainer Jean-Philippe Fortin <jfortin@jhsph.edu>

Description Perform the Adaptive Robust Regression method (ARRm) for the normalization of methylation data from the Illumina Infinium HumanMethylation 450k assay.

License Artistic-2.0

biocViews DNAMethylation, TwoChannel, Preprocessing, Microarray

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

Date/Publication 2026-04-28 12:37:24 UTC

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

RemoteRef HEAD

RemoteSha be3f82451466d6413d73efed624c090004331a55

Contents

ARRmNormalization-package 2
getBackground 2
getCoefficients 3
getDesignInfo 4
getQuantiles 5
normalizeARRm 5
positionPlots 7
ProbesType 8
quantilePlots 8

Index **10**

`ARRmNormalization-package` *ARRm normalization for Illumina methylation data*

Description

Normalize Illumina methylation data from the Infinium HumanMethylation 450k assay with the Adaptive Robust Regression method. The normalization takes care of background intensity, dye bias, chip effects and spatial positions. The normalization can be applied to Beta values, M-values or other metrics as well.

Author(s)

Jean-Philippe Fortin <jfortin@jhsph.edu> Celia M.T. Greenwood <celia.greenwood@mcgill.ca> Aurelie Labbe <aurelie.labbe@mcgill.ca>

`getBackground` *Estimate background intensity from the negative control probes*

Description

This function estimates background intensity for the two colors by taking the median of the negative control probes in each color channel.

Usage

```
getBackground(greenControlMatrix, redControlMatrix)
```

Arguments

`greenControlMatrix`
: matrix of negative control probes intensities in the green channel. Rows are probes, columns are samples.

`redControlMatrix`
: matrix of the negative control probes intensities in the red channel. Rows are probes, columns are samples.

Value

Returns a `data.frame` with two columns; `"green"` contains the background intensity in the green channel for each sample and `"red"` contains the background intensity in the red channel for each sample

Author(s)

Jean-Philippe Fortin <jfortin@jhsph.edu>

Examples

```
data(greenControlMatrix)
data(redControlMatrix)
getBackground(greenControlMatrix,redControlMatrix)
```

---

getCoefficients *Return the coefficients from the ARRm linear model*

Description

For each probe type, it returns the coefficients of the linear model used in the ARRm normalization. Since the model is applied to each percentile separately, different coefficients are returned for every percentile. Residuals are returned as well.

Usage

```
getCoefficients(quantiles,designInfo,backgroundInfo,outliers.perc=0.02)
```

Arguments

quantiles	A list containing three matrices. "$green", "$red" and "$II" must contain respectively the matrices of percentiles obtained from a "betaMatrix" for the Type I Green probes, Type I Red probes and Type II probes. See getQuantiles.
designInfo	matrix returned by getDesignInfo
backgroundInfo	matrix returned by getBackground
outliers.perc	Percentage of outliers to be removed in the regression. By default, set to 0.02

Value

Returns a list containing three lists of coefficients for each probe type. ($green to access coefficients for Type I green probes, $red to access coefficients for Type I red probes and $II to access coefficients for Type II probes). Each list of coefficients contains five subfields. res is a matrix of residuals for the linear model across percentiles (a vector of residuals for each percentile), background.vector is a vector containing the regression coefficients for background intensity across percentiles; dyebias.vector is a vector containing the regression coefficients for dye bias across percentiles; chip.variations is a matrix of chip variations estimated by the linear model; rows correspond to percentiles, columns correspond to chips; position.variations is a matrix of position deviation from the chip mean estimated by the linear model; rows correspond to percentiles, columns correspond to positions.

Author(s)

Jean-Philippe Fortin <jfortin@jhsph.edu>

Examples

```
data(greenControlMatrix)
data(redControlMatrix)
data(sampleNames)
data(betaMatrix)
backgroundInfo=getBackground(greenControlMatrix,redControlMatrix)
designInfo=getDesignInfo(sampleNames)
quantiles=getQuantiles(betaMatrix)
coefficients=getCoefficients(quantiles,designInfo,backgroundInfo)
```

---

getDesignInfo *Build the chip and position indices*

Description

If a vector of sample names of the form "6793856729_R03C02" is given, the function builds a data frame containing chip and position indices for the samples. If no samples names are provided by the user but explicit postion and chip vectors are provided, the data frame is built with these explicit indices.

Usage

```
getDesignInfo(sampleNames = NULL, chipVector = NULL, positionVector = NULL)
```

Arguments

sampleNames	Names of the samples of the form "6793856729_R03C02" (Chip ID, Row, Column)
chipVector	Numeric vector of chip indices (one chip contains 12 samples)
positionVector	Numeric vector of on-chip position indices (between 1 and 12)

Value

A `data.frame` containing a column named `chipInfo` containing the chip indices, a column named `positionInfo` containing the position indices, and a column `sampleNames` if sample names were provided.

Author(s)

Jean-Philippe Fortin <jfortin@jhsph.edu>

Examples

```
data(sampleNames)
getDesignInfo(sampleNames)
```

getQuantiles *Return the percentiles of a betaMatrix for each probe type*

Description

It returns the percentiles of a betaMatrix for Type I Green, Type I Red and Type II probes. If no list of probes is provided, all probes are taken into account to compute the percentiles.

Usage

```
getQuantiles(betaMatrix,goodProbes=NULL)
```

Arguments

betaMatrix	matrix containing the Beta values. Rows are probes, columns are samples.
goodProbes	Ids of the probes to be normalized (Id. of the form "cg00000029").

Value

Returns a list of three matrices of percentiles. For Type I green and Type I red probes, the corresponding matrices can be accessed by $green and $red. For Type II probes, the matrix can be accessed by $II

Author(s)

Jean-Philippe Fortin <jfortin@jhsph.edu>

Examples

```
data(greenControlMatrix)
data(redControlMatrix)
data(sampleNames)
data(betaMatrix)
quantiles=getQuantiles(betaMatrix)
```

normalizeARRm *Perform ARRm normalization*

Description

This function perform Adaptive Robust Regression method (ARRm) normalization on Beta values. The method corrects for background intensity, dye bias and spatial on-chip position. By default, chip mean correction is also performed.

Usage

```
normalizeARRm(betaMatrix, designInfo, backgroundInfo, outliers.perc = 0.02,
goodProbes = NULL,chipCorrection=TRUE)
```

Arguments

betaMatrix	matrix containing the Beta values. Rows are probes, columns are samples.
designInfo	A data.frame containing a column named chipInfo containing the chip indices and a column named positionInfo containing the position indices
backgroundInfo	A data.frame containing two columns: green contains the background intensity in the green channel for each sample and red contains the background intensity in the red channel for each sample
outliers.perc	Proportion (between 0 and 1) of outliers to be removed from the ARRm regression
goodProbes	Ids of the probes to be normalized (Id. of the form "cg00000029")
chipCorrection	logical, should normalization correct for chip mean?

Value

A matrix containing the normalized Beta values

Author(s)

Jean-Philippe Fortin <jfortin@jhsph.edu>

See Also

getBackground to see how to obtain background information from control probes, and getDesignInfo to see how to obtain position and chip indices

Examples

```
data(greenControlMatrix)
data(redControlMatrix)
data(sampleNames)
data(betaMatrix)
backgroundInfo=getBackground(greenControlMatrix, redControlMatrix)
designInfo=getDesignInfo(sampleNames)
normMatrix=normalizeARRm(betaMatrix, designInfo, backgroundInfo, outliers.perc = 0.02)
```

positionPlots *Plots to evalue chip position effects on different percentiles*

Description

For each probe type, and for each sample, deviations from the chip mean are computed for a given percentile. These deviations are plotted against on-chip position.

Usage

```
positionPlots(quantiles,designInfo,percentiles=c(25,50,75))
```

Arguments

quantiles	A list containing three matrices. list$green, list$red and list$II must contain respectively the matrices of percentiles obtained from a betaMatrix for the Type I Green probes, Type I Red probes and Type II probes. See getQuantiles.
designInfo	designInfo matrix returned by getDesignInfo
percentiles	Vector of percentiles to be plotted. By default, the 25th, 50th and 75th percentiles are plotted. (percentiles=c(25,50,75)).

Value

Plots are produced and saved as pdf in the current directory.

Author(s)

Jean-Philippe Fortin <jfortin@jhsph.edu>

Examples

```
data(greenControlMatrix)
data(redControlMatrix)
data(sampleNames)
data(betaMatrix)
quantiles=getQuantiles(betaMatrix)
backgroundInfo=getBackground(greenControlMatrix, redControlMatrix)
designInfo=getDesignInfo(sampleNames)
positionPlots(quantiles, designInfo, percentiles=c(25,50,75))
```

ProbesType *Probe Design information for the 450k methylation assay*

Description

Probe Design information for the Illumina Infinium HumanMethylation 450k array. To each probe is associated the design type, either Infinium I Green, Infinium I Red or Infinium II. Probe names follows Illumina's annotation (names of the form "cg00000029").

Usage

```
data(ProbesType)
```

Format

A data frame containing two columns. `$Probe_Name` contains the names of the probes, and `$Design_Type` contains the design information ("I Green", "I Red" or "II").

Examples

```
data(ProbesType)
```

quantilePlots *Diagnostic plots for evaluation of background effects and dye bias effects on different percentiles*

Description

For each probe type, and for each sample, several percentiles are plotted against background intensity, and also against dye bias.

Usage

```
quantilePlots(quantiles,backgroundInfo,designInfo,percentilesI=NULL,percentilesII=NULL)
```

Arguments

`quantiles`	A `list` containing three matrices. `list$green`, `list$red` and `list$II` must contain respectively the matrices of percentiles obtained from a `betaMatrix` for the Type I Green probes, Type I Red probes and Type II probes. See `getQuantiles`.
`designInfo`	`designInfo` matrix returned by `getDesignInfo`
`backgroundInfo`	`"backgroundInfo"` matrix returned by `getBackground`
`percentilesI`	List of percentiles to be plotted for Type I probes. Must be a vector of integers from 1 to 100. If set to `NULL` (by default), the sequence (5,10,...,95) of percentiles is plotted.

`percentilesII` List of percentiles to be plotted for Type II probes. Must be a vector of integers from 1 to 100. If set to `NULL` (by default), the sequence (10,20,...,90) of percentiles is plotted.

Value

Plots are produced and saved as pdf in the current directory.

Author(s)

Jean-Philippe Fortin <jfortin@jhsph.edu>

Examples

```
data(greenControlMatrix)
data(redControlMatrix)
data(sampleNames)
data(betaMatrix)
quantiles=getQuantiles(betaMatrix)
backgroundInfo=getBackground(greenControlMatrix, redControlMatrix)
designInfo=getDesignInfo(sampleNames)
quantilePlots(quantiles, backgroundInfo, designInfo)
```

Index

∗ **package**
ARRmNormalization-package, 2

ARRmNormalization
*(*ARRmNormalization-package*)*, 2
ARRmNormalization-package, 2

getBackground, 2, *3*, *6*, *8*
getCoefficients, 3
getDesignInfo, *3*, 4, *6–8*
getQuantiles, *3*, 5, *7, 8*

normalizeARRm, 5

positionPlots, 7
ProbesType, 8

quantilePlots, 8