Package ‘wateRmelon’

May 17, 2024

**Type** Package

**Title** Illumina 450 and EPIC methylation array normalization and metrics

**Version** 2.10.0

**Description** 15 flavours of betas and three performance metrics, with methods for objects produced by methylumi and minfi packages.

**License** GPL-3

**Depends** R (>= 3.5.0), Biobase, limma, methods, matrixStats, methylumi, lumi, ROC, IlluminaHumanMethylation450kanno.ilm12.hg19, illuminaio

**Imports** Biobase

**Enhances** minfi


**LazyLoad** yes

**biocViews** DNAMethylation, Microarray, TwoChannel, Preprocessing, QualityControl

pwod.R readEPIC.R preprocessIlluminaMethylation.R
robustQuantileNorm_Illumina450K.probeCategories.R
swan2.R uniqueAnnotationCategory.R qual.R uSexQN.R smokp.R
z_bigmelon.R

RoxygenNote 7.3.1

NeedsCompilation no

VignetteBuilder knitr

Encoding UTF-8

git_url https://git.bioconductor.org/packages/wateRmelon

git_branch RELEASE_3_19

git_last_commit 0e0ae03

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-16

Author Leo C Schalkwyk [cre, aut],
       Tyler J Gorrie-Stone [aut],
       Ruth Pidsley [aut],
       Chloe CY Wong [aut],
       Nizar Touleimat [ctb],
       Matthieu Defrance [ctb],
       Andrew Teschendorff [ctb],
       Jovana Maksimovic [ctb],
       Louis Y El Khoury [ctb],
       Yucheng Wang [ctb],
       Alexandria Andrayas [ctb]

Maintainer Leo C Schalkwyk <lschal@essex.ac.uk>

Contents

wateRmelon-package ...................................................... 3
.createAnnotation ......................................................... 4
.getManifestString ........................................................ 4
adaptRefQuantiles ........................................................ 5
adjustedDasen ............................................................. 6
adjustedFunnorm .......................................................... 7
agep .............................................................. 8
as.methylumi-methods ..................................................... 9
beadc ............................................................ 10
beadcount ............................................................. 11
Beta2M ............................................................ 12
betaqn-exprmethy450-methods ........................................... 12
Description

Functions for calculating the index of DNA methylation proportion beta in 15 different ways, and three different ways of estimating data quality or normalization performance.
Details

Package: wateRmelon
Type: Package
Version: 1.0
Date: 2012-10-10
License: GPL3

Author(s)

Leonard C Schalkwyk, Ruth Pidsley and Chloe Wong
Maintainer: Who to complain to <leonard.schalkwyk@kcl.ac.uk>

References


Description

Internal function to guess correct pData and retrieve using minfi

Usage

.createAnnotation(object)

Description

Internal functions for Illumina i450 normalization functions

Usage

.getManifestString(annotation)
adaptRefQuantiles

Arguments

- `x` a MethylumiSet
- `obj` a MethylSet
- `fd` a character vector of the desired annotation columns
- `rn` a character vector of the desired features

Details

got returns a character vector of 'I' and 'II', fot returns the index of the relevant column. pop returns a data frame

Author(s)

lschal@essex.ac.uk

References


adaptRefQuantiles  Functions from 450-pipeline (Touleimat & Tost)

Description

These functions are part of the 450K pipeline (Touleimat and Tost, Epigenomics 2012 4:325). For freestanding use of the normalization function, a wrapper is provided, see tost

Author(s)

Nizar Touleimat, wrapper by Leonard.Schalkwyk@kcl.ac.uk

References


**adjustedDasen**

---

**Description**

adjustedDasen utilizes dasen normalization to normalize autosomal CpGs, and infers the sex chromosome linked CpGs by linear interpolation on corrected autosomal CpGs.

**Usage**

```r
adjustedDasen(
  mns,
  uns,
  onetwo,
  chr,
  offset_fit = TRUE,
  cores = 1,
  ret2 = FALSE,
  fudge = 100,
  ...
)
```

**Arguments**

- **mns**: matrix of methylated signal intensities, samples in column and probes in row.
- **uns**: matrix of unmethylated signal intensities, samples in column and probes in row.
- **onetwo**: character vector or factor of length nrow(mns) indicating assay type 'I' or 'II'.
- **chr**: character vector stores the mapped chromosomes for all probes, e.g. `chr <- c('1', 'X', '21', 'Y')`.
- **offset_fit**: logical (default is TRUE). To use dasen, set it TRUE; to use nasen, set it FALSE.
- **cores**: an integer (e.g. 8) defines the number of cores to parallel processing. Default value is 1, set to -1 to use all available cores.
- **ret2**: logical (default is FALSE), if TRUE, returns a list of intensities and betas instead of a naked matrix of betas.
- **fudge**: default 100, a value added to total intensity to prevent denominators close to zero when calculating betas, e.g. `betas <- mns / (mns + uns + fudge)`.
- **...**: additional argument roco for dfsfit giving Sentrix rows and columns. This allows a background gradient model to be fit. This is split from data column names by default. `roco=NULL` disables model fitting (and speeds up processing), otherwise roco can be supplied as a character vector of strings like 'R01C01' (only 3rd and 6th characters used).

**Value**

a matrix of normalised beta values.
**adjustedFunnorm**

**References**

A data-driven approach to preprocessing Illumina 450K methylation array data, Pidsley et al, BMC Genomics.


**Examples**

```r
data(melon)
normalised_betas <- adjustedDasen(mns = methylated(melon), uns = unmethylated(melon), onetwo = fData(melon)[, fot(melon)]
## if input is an object of methylumiset or methylset
normalised_betas <- adjustedDasen(melon)
```

---

**Description**

adjustedFunnorm utilizes functional normlisation to normalise autosomal CpGs, and infers the sex chromosome linked CpGs by linear interpolation on corrected autosomal CpGs.

**Usage**

```r
adjustedFunnorm(
  rgSet,
  nPCs = 2,
  sex = NULL,
  bgCorr = TRUE,
  dyeCorr = TRUE,
  keepCN = TRUE,
  ratioConvert = TRUE,
  verbose = TRUE
)
```

**Arguments**

- **rgSet**: An object of class "RGChannelSet".
- **nPCs**: Number of principal components from the control probes PCA.
- **sex**: An optional numeric vector containing the sex of the samples.
- **bgCorr**: Should the NOOB background correction be done, prior to functional normalization (see "preprocessNoob")
- **dyeCorr**: Should dye normalization be done as part of the NOOB background correction (see "preprocessNoob")?
- **keepCN**: Should copy number estimates be kept around? Setting to 'FALSE' will decrease the size of the output object significantly.
ratioConvert Should we run "ratioConvert", ie. should the output be a "GenomicRatioSet" or should it be kept as a "GenomicMethylSet"; the latter is for experts.

verbose Should the function be verbose?

Value

an object of class "GenomicRatioSet", unless "ratioConvert=FALSE" in which case an object of class "GenomicMethylSet".

References

Functional normalization of 450k methylation array data improves replication in large cancer studies, Fortin et al., 2014, Genome biology.

Examples

## Not run:
GRset <- adjustedFunnorm(RGSet)
## End(Not run)

agep Age Prediction from methylomic expression data

Description

Predict age of samples using Horvaths Coefficients

Usage

agep(betas, coeff = NULL, method = c('horvath', 'hannum', 'phenoage', 'skinblood', 'lin', 'all'), n_missing = TRUE, missing_probes = FALSE, ...)

Arguments

betas Matrix of betas or MethyLumiSet or MethylSet object.
coeff If NULL, will default to whatever method is specified in method. If not NULL, the expected input should be a vector of coefficients and intercept
method Currently: "horvath", "hannum", "phenoage", "skinblood", "lin" and "all", if "all" agep will seek to calculate ages using all methods else will use the method specified. Default is "horvath".
n_missing Logical, additionally output the number of missing CpGs for each sample using the specified method or coeff list.
missing_probes Logical, additionally output the names of missing CpGs for each sample using the specified method or coeff list.
... To pass to arguments to downstream functions to specify adult.age
Value

Returns a matrix of predicted ages per sample. With additional columns created whether `n_missing` or `missing_probes` are specified. If method is "all" then all ages will be provided in the same matrix output.

Author(s)

Original Functions: Steve Horvath
wateRmelon Implementation: Tyler Gorrie-Stone, Leo Schalkwyk, Louis El Khoury

References

Horvath S: DNA methylation age of human tissues and cell types. Genome Biology 2013, 14:R115

Examples

data(melon)
agep(melon, coeff=NULL, method="all", n_missing=FALSE)
agep(melon, coeff=NULL, method="horvath", n_missing=TRUE)

Description

Returns a MethyLumiSet object populated with the data provided. There are MethyLumiSet and MethylSet methods. In the default method, the data is all optional. Please note that for the results to be sane, `mn`, `un`, `bn`, and `pv` have to be in the same sample and feature order and the same size. The function does not currently do any checks!

Usage

# default method
as.methylumi (mn = NULL, un = NULL, bn = NULL, pv = NULL, qc = NULL, da = NULL,...)

Arguments

- **mn**: matrix of methylated signal intensities, each column representing a sample (generic) or a MethyLumiSet, RGSet, or MethylSet object. Column names are used to get Sentrix row and column by default, see `...`
- **un**: matrix of unmethylated signal intensities, each column representing a sample (default method) or NULL when `mn` is an object containing methylated and unmethylated values
- **bn**: matrix of precalculated betas, each column representing a sample
- **pv**: matrix of detection p-values, each column representing a sample
annotation data frame, such as `x@featureData@data` #methylumi package. If NULL (the default), the `IlluminaHumanMethylation450kmanifest` package is used. See the `fd` argument

control probe intensities: list of 2 matrices, Cy3 and Cy5, with rownames, such as produced by `intensitiesByChannel(QCdata(x))` (methylumi package)

Other arguments such as a `featureData` object or optional `assayData`

**Methods**

signature(mn = "MethylSet") Coerces a `MethylSet` to a `MethyLumiSet`, and provides it with a set of `featureData`, which by default is just the chromosome and DESIGN (ie type I or type II assay). Other data can be included using the `fd` argument, available data is listed by the function `getColumns()`

signature(mn = "MethyLumiSet") This is mainly useful for adding `featureData` as described under `MethylSet` above. MethyLumiSet objects produced by methylumiR have the full annotation, those from methylumIDAT do not, and functions such as `swan` require it

signature(mn = "ANY") `as.methylumi` (mn = NULL, un = NULL, bn = NULL, pv = NULL, qc = NULL, da = NULL, fd = c("CHR", "DESIGN"))

---

**Description**

Calculates the number of samples with bead count <3 for each probe in matrix of bead count values.

**Usage**

`beadc(x)`

**Arguments**

- `x` matrix of bead count values returned by the `beadcount` function

**Value**

Vector of number of samples with bead count <3 for each probe

**Note**

The `beadc` function is internal to the `pfilter` function

**Author(s)**

ruth.pidsley@kcl.ac.uk
beadcount

References


beadcount

Creates matrix of beacounts from minfi data.

Description

Creates matrix of beacounts from data read in using the minfi package. NAs represent probes with beadcount <3. An Extended RG Channel Set is required for this function to work.

Usage

beadcount(x)

Arguments

x 450K methylation data read in using minfi to create an Extended RG Channel Set

Value

A matrix of bead counts with bead counts <3 represented by NA for use in the pfilter function for quality control

Note

The beadcount function is internal to the pfilter function

Author(s)

Ruth.Pidsley@kcl.ac.uk

References

betaqn-exrnmethy450-methods

---

**Beta2M**

*Internal functions for peak.correction (fuks)*

---

**Description**

Internal functions for peak.correction

**Usage**

\[ \text{Beta2M}(B) \]

**Arguments**

- \( B \): a vector or matrix of beta values for conversion

**Value**

a vector or matrix of the same shape as the input

**Author(s)**

Matthieu Defrance <defrance@bigre.ulb.ac.be>

**References**


---

**betaqn-exrnmethy450-methods**

*Calculate normalized betas from exprmethy450 of Illumina 450K methylation arrays*

---

**Description**

Quantile normalize betas from exprmethy450 objects

**Arguments**

- \( bn \): An exprmethy450 object.
- \( \text{fudge} \): value added to total intensity to prevent denominators close to zero when calculating betas

**Details**

**betaqn** quantile normalizes betas
Value

exprmethy450 object of the same shape and order as bn.

Author(s)

Leonard.Schalkwyk@kcl.ac.uk

References


**BMIQ**

Beta-Mixture Quantile (BMIQ) Normalisation method for Illumina 450k arrays

Description

BMIQ is an intra-sample normalisation procedure, correcting the bias of type-2 probe values. BMIQ uses a 3-step procedure: (i) fitting of a 3-state beta mixture model, (ii) transformation of state-membership probabilities of type2 probes into quantiles of the type1 distribution, and (iii) a conformal transformation for the hemi-methylated probes. Exact details can be found in the reference below.

Usage

```r
BMIQ(beta.v, design.v, nL = 3, doH = TRUE, nfit = 50000, th1.v = c(0.2, 0.75), th2.v = NULL, niter = 5, tol = 0.001, plots = TRUE, sampleID = 1, pri=TRUE)
```

Arguments

- **beta.v**: vector consisting of beta-values for a given sample, or a MethyLumiSet or MethylSet containing multiple samples. For the MethyLumiSet and MethylSet methods, this is the only required argument, and the function will be run on each sample.

- **design.v**: corresponding vector specifying probe design type (1=type1,2=type2). This must be of the same length as beta.v and in the same order.
nL  number of states in beta mixture model. 3 by default. At present BMIQ only works for nL=3.

doH perform normalisation for hemimethylated type2 probes. These are normalised using an empirical conformal transformation and also includes the left-tailed type2 methylated probes since these are not well described by a beta distribution. By default TRUE.

nfit number of probes of a given design type to use for the fitting. Default is 50000. Smaller values (~10000) will make BMIQ run faster at the expense of a small loss in accuracy. For most applications, 5000 or 10000 is ok.

th1.v thresholds used for the initialisation of the EM-algorithm, they should represent best guesses for calling type1 probes hemi-methylated and methylated, and will be refined by the EM algorithm. Default values work well in most cases.

th2.v thresholds used for the initialisation of the EM-algorithm, they should represent best guesses for calling type2 probes hemi-methylated and methylated, and will be refined by the EM algorithm. By default this is null, and the thresholds are estimated based on th1.v and a modified PBC correction method.

niter maximum number of EM iterations to do. This number should be large enough to yield good fits to the type1 distribution. By default 5.

tol tolerance convergence threshold for EM algorithm. By default 0.001.

plots logical specifying whether to plot the fits and normalised profiles out. By default TRUE.

sampleID the ID of the sample being normalised.

pri logical: print verbose progress information?

pnbeta.v BMIQ normalised profile.

Details

Full details can be found in the reference below. Note: these functions require the RPMM package, not currently a dependency of the wateRmelon package.

Value

Default method: A list with following entries:

nbeta the normalised beta-profile for the sample
class1 the assigned methylation state of type1 probes
class2 the assigned methylation state of type2 probes
av1 the mean beta-values for the nL states for type1 probes
av2 the mean beta-values for the nL states for type2 probes
hf the estimated "Hubble" dilation factor used in the normalisation of hemi-methylated probes
th1 estimated thresholds for calling unmethylated and methylated type1 probes
th2 estimated thresholds for calling unmethylated and methylated type2 probes

MethyLumiSet method: A methyLumiSet object
bscon

Author(s)
Andrew Teschendorff, MethyLumiSet method by Leo Schalkwyk Leonard.Schalkwyk@kcl.ac.uk

References

Examples
# library(RPMM)
# data(melon)
# BMIQ(melon,nfit=100)

bscon

Description
Uses control data from Infinium HumanMethylation450 BeadChip to calculate bisulfite conversion for each array

Usage
bscon(x, ...) # S4 methods exist for RGChannelSet and MethyLumiset objects

Arguments
x IDAT or report files containing 450k data
... current methods have no optional arguments

Details
This function uses the green and red channels reading of the type I and type II bisulfite conversion data to return the median bisulfite conversion percentage value for each array.

For the type I chemistry the beta values are calculated by dividing the first three probes of the green channel (C1, C2, C3) and the second three probes of the red channel (C4, C5, C6) by the sum of these probes and the unconverted probes of the green (U1, U2, U3) and the red (U4, U5, U6) channel.

The beta values from type II chemistry are calculated by dividing the methylated (red) channels by the sum of methylated (red) and unmethylated (green) channels.

Value
A vector of percentage values referring to the bisulfite conversion levels of each array.
Note

Updates to HumanMethylationEPIC manifest has seen the removal of control probes C6 and U6. This does not appear to grossly affect how function performs however we are considering alternative approaches to account for this.

Author(s)

Louis El Khoury (louis.el-khoury@essex.ac.uk), Eilis Hannon, Leonard Schalkwyk (lschal@essex.ac.uk)

Examples

```r
library(wateRmelon)
data(melon)
bs <- bscon(melon)
bs
```

canno

*canno* - process csv manifest into annotation object for illumina methylation preprocessing

Description

canno - process csv manifest into annotation object for illumina methylation preprocessing

Usage

canno(man = "EPIC-8v2-0_A1.csv", name = NULL)

Arguments

* man
* name

Details

This is based on the scripts shipped with minfi annotation packages. It is based on existing Illumina Human Methylation csv format manifests, but because reverse-engineered, may require updates to work on future products.

Value

IlluminaMethylationManifest
Description

Methods for function \texttt{colnames} in package \texttt{wateRmelon}.

Methods:

\texttt{signature(x = "MethyLumiSet")} returns the sample names


description

This is a wrapper for combining different MethyLumiSet objects.

Usage

\texttt{combo(...)}

Arguments

\texttt{...} Eventually, any number of MethyLumiSet objects. Currently only guaranteed for 2 objects.

Details

This is a wrapper for \texttt{methylumi::combine}, which works around a name clash with a different combine function from the \texttt{gdata} package, and also a bug in \texttt{methylumi::combine}.

Value

a MethyLumiSet. The assayData, QCdata, experimentData, protocolData and phenoData are joined on \texttt{sampleName}. featureData and annotation are taken from the object given in the first argument.

Note

the function uses sampleNames and gets rid of duplicates. Numeric sampleNames cause problems (and are a Bad Idea anyway). They should be turned into names with \texttt{make.names()} first.

Author(s)

Leo Schalkwyk <leonard.schalkwyk@kcl.ac.uk>
Dasen

Calculate normalized betas from Illumina 450K methylation arrays

Description

Multiple ways of calculating the index of methylation (beta) from methylated and unmethylated probe intensities used in Pidsley et al 2012. S4 methods exist where possible for MethyLumiSet, MethylSet, RGSet and exprmeth450 objects.

Usage

dasen ( mns, uns, onetwo, fudge = 100, ret2=FALSE, ... )
nasen ( mns, uns, onetwo, ret2=FALSE, fudge = 100, ... )
betaq(Q ( bn )
naten ( mn, un, fudge = 100, ret2=FALSE, ... )
nanet ( mn, un, fudge = 100, ret2=FALSE, ... )
nanes ( mns, uns, onetwo, fudge = 100, ret2=FALSE, ... )
danes ( mn, un, onetwo, fudge = 100, ret2=FALSE, ... )
danet ( mn, un, onetwo, fudge = 100, ret2=FALSE, ... )
daten ( mns, uns, onetwo, fudge = 100, ret2=FALSE, ... )
daten2 ( mn, un, onetwo, fudge = 100, ret2=FALSE, ... )
tost ( mn, un, da, pn )
fuks ( data, anno)
swan ( mn, un, qc, da=NULL, return.MethylSet=FALSE )
Arguments

- **mn, mns**: matrix of methylated signal intensities, each column representing a sample (generic) or a MethylLumiSet, RGSet, or MethylSet object. Column names are used to get Sentrix row and column by default, see ‘...’.
- **un, uns**: matrix of unmethylated signal intensities, each column representing a sample (default method) or NULL when mn is an object containing methylated and unmethylated values.
- **bn, data**: matrix of precalculated betas, each column representing a sample.
- **onetwo**: character vector or factor of length nrow(mn) indicating assay type 'I' or 'II'.
- **pn**: matrix of detection p-values, each column representing a sample.
- **da, anno**: annotation data frame, such as x@featureData@data #methylumi package. If NULL, the swan method requires the IlluminaHumanMethylation450kmanifest package.
- **qc**: control probe intensities: list of 2 matrices, Cy3 and Cy5, with rownames, such as produced by intensitiesByChannel(QCdata(x)) #methylumi package.
- **fudge**: value added to total intensity to prevent denominators close to zero when calculating betas.
- **return.MethylSet**: if TRUE, returns a MethylSet object instead of a naked matrix of betas.
- **ret2**: if TRUE, returns a list of intensities and betas instead of a naked matrix of betas.
- **...**: additional argument roco for dfsfit giving Sentrix rows and columns. This allows a background gradient model to be fit. This is split from data column names by default. roco=NULL disables model fitting (and speeds up processing), otherwise roco can be supplied as a character vector of strings like 'R01C01' (only 3rd and 6th characters used).

Details

- **dasen**: same as nasen but type I and type II backgrounds are equalized first. This is our recommended method.
- **betaqn**: quantile normalizes betas.
- **naten**: quantile normalizes methylated and unmethylated intensities separately, then calculates betas.
- **nanet**: quantile normalizes methylated and unmethylated intensities together, then calculates betas. This should equalize dye bias.
- **nanes**: quantile normalizes methylated and unmethylated intensities separately, except for type II probes where methylated and unmethylated are normalized together. This should equalize dye bias without affecting type I probes which are not susceptible.
- **danes**: same as nanes, except type I and type II background are equalized first.
- **danet**: same as nanet, except type I and type II background are equalized first.
- **dalen**: background equalization only, no normalization.
- **daten1**: same as naten, except type I and type II background are equalized first (smoothed only for methylated).
**daten2** same as naten, except type I and type II background are equalized first (smoothed for methylated an unmethylated)

**nansen** same as naten but type I and type II intensities quantile normalized separately

**tost** method from Touleimat and Tost 2011

**fusks** method from Dedeurwaerder et al 2011. Peak correction only, no normalization

**swan** method from Maksimovic et al 2012

**Value**

a matrix (default method) or object of the same shape and order as the first argument containing betas.

**Author(s)**

Leonard.Schalkwyk@kcl.ac.uk

**References**


**See Also**

*pfilter, as.methylumi*

**Examples**

#MethyLumiSet method
data(melon)
melon.dasen <- dasen(melon)
Calculation of normalized betas from MethyLumiSets of Illumina 450K methylation arrays

Description

Multiple ways of calculating the index of methylation (beta) from methylated and unmethylated probe intensities used in Pidsley et al 2012.

Arguments

- \textbf{mn, mns, data, bn} A MethyLumiSet object. Sample names names are used to get Sentrix row and column by default, see ‘...’.
- \textbf{fudge} value added to total intensity to prevent denominators close to zero when calculating betas
- \textbf{...} additional argument roco for dfsfit giving Sentrix rows and columns. This allows a background gradient model to be fit. This is split from data column names by default. roco=\texttt{NULL} disables model fitting (and speeds up processing), otherwise roco can be supplied as a character vector of strings like 'R01C01' (only 3rd and 6th characters used).

Details

- \textbf{dasen} same as nasen but type I and type II backgrounds are normalized first. This is our recommended method
- \textbf{betaqn} quantile normalizes betas
- \textbf{naten} quantile normalizes methylated and unmethylated intensities separately, then calculates betas
- \textbf{nanet} quantile normalizes methylated and unmethylated intensities together, then calculates betas. This should equalize dye bias.
- \textbf{nanes} quantile normalizes methylated and unmethylated intensities separately, except for type II probes where methylated and unmethylated are normalized together. This should equalize dye bias without affecting type I probes which are not susceptible.
- \textbf{danen} same as nanes, except type I and type II background are equalised first.
- \textbf{danet} same as nanet, except type I and type II background are equalised first.
- \textbf{danen} background equalisation only, no normalization
- \textbf{daten1} same as naten, except type I and type II background are equalised first (smoothed only for methylated)
- \textbf{daten2} same as naten, except type I and type II background are equalised first (smoothed for methylated an unmethylated)
- \textbf{nasen} same as nasen but type I and type II intensities quantile normalized separately
- \textbf{tost} method from Touleimat and Tost 2011
- \textbf{fuks} method from Dedeurwaerder et al 2011. Peak correction only, no normalization
- \textbf{swan} method from Maksimovic et al 2012
**Value**

a matrix (default method) or object of the same shape and order as the first argument containing betas.

**methods**

dasen (mns, fudge = 100, ... ) nasen (mns, fudge = 100) betaqn(bn) naten (mn, fudge = 100) nanes (mns, fudge = 100) danes (mn, fudge = 100, ...) danet (mn, fudge = 100, ...) daten1(mn, fudge = 100, ...) daten2(mn, fudge = 100, ...) tost (mn) fuks (data) swan (mn)

**Author(s)**

Leonard.Schalkwyk@kcl.ac.uk

**References**


**Calculate normalized betas from Illumina 450K methylation arrays**

**Description**

Multiple ways of calculating the index of methylation (beta) from methylated and unmethylated probe intensities used in Pidsley et al 2012.

**Arguments**

- **mn, mns**
  matrix of methylated signal intensities, each column representing a sample (default method), or an object for which a method is available. Column names are used to get Sentrix row and column by default, see ‘...’.

- **un, uns**
  matrix of unmethylated signal intensities, each column representing a sample (default method) or NULL when mn is an object containing methylated and unmethylated values

- **bn, data**
  matrix of precalculated betas, each column representing a sample

- **onetwo**
  character vector or factor of length nrow(mn) indicating assay type ‘I’ or ‘II’
da, anno annotation data frame, such as x@featureData@data #methylumi package
qc control probe intensities: list of 2 matrices, Cy3 and Cy5, with rownames, such as produced by intensitiesByChannel(QCdata(x)) #methylumi package
fudge value added to total intensity to prevent denominators close to zero when calculating betas

... additional argument roco for dfsfit giving Sentrix rows and columns. This allows a background gradient model to be fit. This is split from data column names by default. roco=NULL disables model fitting (and speeds up processing), otherwise roco can be supplied as a character vector of strings like ’R01C01’ (only 3rd and 6th characters used).

Details

dasen same as nasen but type I and type II backgrounds are normalized first. This is our recommended method
betaqn quantile normalizes betas
naten quantile normalizes methylated and unmethylated intensities separately, then calculates betas
nanet quantile normalizes methylated and unmethylated intensities together, then calculates betas. This should equalize dye bias.
nanes quantile normalizes methylated and unmethylated intensities separately, except for type II probes where methylated and unmethylated are normalized together. This should equalize dye bias without affecting type I probes which are not susceptible.
danes same as nanes, except type I and type II background are equalised first.
danet same as nanet, except type I and type II background are equalised first.
danen background equalisation only, no normalization
daten1 same as naten, except type I and type II background are equalised first (smoothed only for methylated)
daten2 same as naten, except type I and type II background are equalised first (smoothed for methylated an unmethylated)
nasen same as naten but type I and type II intensities quantile normalized separately
tost method from Touleimat and Tost 2011
fuks method from Dedeurwaerder et al 2011. Peak correction only, no normalization
swan method from Maksimovic et al 2012

Value
a matrix of betas is returned by the MethySet and RGChannelSet methods because they do not have a defined slot for betas.

methods

dasen ( mns, uns, onetwo, fudge = 100, ... ) nasen ( mns, uns, onetwo, fudge = 100 ) betaqn( bn ) naten ( mn, un, fudge = 100 ) nanet ( mn, un, fudge = 100 ) nanet ( mn, un, fudge = 100 ) nanes ( mns, uns, onetwo, fudge = 100 ) danes ( mn, un, onetwo, fudge = 100, ... ) danet (
### db1

**Internal watermelon functions for calculating betas**

#### Description

`db1` is used for quantile normalizing methylated together with unmethylated (dye bias methods) data. `dansen` functions are used for smoothing the background equalization in methods whose names start with `d` (daten etc).

#### Usage

```r
db1(mn, un)
dfsfit(mn, onetwo, roco=substring(colnames(mn), regexpr("R0[1-9]C0[1-9]", colnames(mn))), ...)
dfs2(x, onetwo)
```

#### Arguments

- **mn, x**: matrix of methylated signal intensities, each column representing a sample (default method), or an object for which a method is available. For `dfsfit` and `dfs2`, this can also be a matrix of unmethylated intensities.
- **un**: matrix of unmethylated signal intensities, each column representing a sample (default method) or NULL when `mn` is an object containing methylated and unmethylated values.
- **onetwo**: character vector or factor of length `nrow(mn)` indicating assay type 'I' or 'II'.

---

**References**


\textbf{dmrse}

\textbf{roco} \quad roco for dfsfit giving Sentrix rows and columns. This allows a background gradient model to be fit. This is split from data column names by default. roco=NULL disables model fitting (and speeds up processing), otherwise roco can be supplied as a character vector of strings like ’R01C01’ (3rd and 6th characters used).

\ldots \quad \text{no additional arguments currently used}

\textbf{Details}

db1 - quantile normalizes methylated against unmethylated (basic function for dyebuy* dye bias methods). dfsfit - corrects the difference in backgrounds between type I and type II assays and fits a linear model to Sentrix rows and columns if these are available to improve precision where there is a background gradient. dfs2 - finds the difference between type I and type II assay backgrounds for one or more samples.

\textbf{Value}

db1 - a list of 2 matrices of intensities, methylated and unmethylated
dfsfit - a matrix of adjusted intensities
dfs2 - a background offset value

\textbf{Author(s)}

Leonard.Schalkwyk@kcl.ac.uk

\textbf{References}


\textbf{dmrse} \quad \textit{Standard error of iDMR 450k array DNA methylation features}

\textbf{Description}

Imprinting differentially methylated regions (iDMRs) are expected to be approximately half methylated, as is observed at the 227 probes in known iDMRs. These functions calculate measures of dispersion for the beta values at these CpG sites, of which the most useful is dmrse_row, which is the between-sample standard error.

\textbf{Usage}

dmrse(betas, idmr = iDMR())
dmrse_col(betas, idmr = iDMR())
dmrse_row(betas, idmr = iDMR())
Arguments

betas      a matrix of betas (default method), a MethyLumiSet object (methylumi package), a MethylSet or RGChannelSet object (minfi package) or a exprmethy450 object (IMA package).

idmr      a character vector of iDMR probe names such as returned by iDMR()

Value

return a standard error of the mean of betas for all samples and iDMR probes (dmrse) or the standard error of the mean for just the between sample component(dmrse_row) or between probe(dmrse_col) component.

Author(s)

Leonard.Schalkwyk@kcl.ac.uk

References


See Also

seabi, a sex-difference metric, and genki, based on SNPs.

Examples

#MethyLumiSet method
data(melon)
dmrse(melon)

#MethyLumiSet method after normalization
melon.dasen <- dasen(melon)
dmrse(melon.dasen)

Methods for Function dmrse in Package wateRmelon

Description

Methods for function dmrse, dmrse_row and dmrse_col in package wateRmelon. Please see dmrse for details of the calculation of this standard-error performance metric.

Methods:

signature(betas = "exprmethy450") all of the methods simply extract betas from the data object (which can be a exprmethy450, MethylSet, MethyLumiSe, or RGChannelSet) and calculate the metric.
epicv2clean.default  Strip and subset EPICv2 data to work with legacy data and methods

Description

Returns an object with rownames stripped of the EPICv2 suffixes, duplicate probes are omitted.

Usage

```r
## Default S3 method:
epicv2clean(x)
```

Details

EPICv2 manifests contain a few thousand probes with up to 10 replicate syntheses. To accommodate this a modified naming scheme is used, so none of the probe names match those on the EPIC and previous arrays (even though most of the probes are the same sequence and presumably similar performance).

This simple function relies on the rowname and subsetting methods and will work for matrix, dataframe, MethyLumiSet, or MethylSet objects, and there is a method for gds (bigmelon) objects.

estimateCellCounts  Cell Proportion Estimation using watermelon

Description

Estimates relative proportion of pure cell types within a sample, mostly identical to `estimateCellCounts`. References for both 450k and EPIC array are available. However 450k reference can be used on EPIC data by specifying the reference platform. Additionally a measure of error is calculated as a means of quality control.

Usage

```r
estimateCellCounts.wmln(
  object,
  referencePlatform = c("IlluminaHumanMethylation450k", "IlluminaHumanMethylationEPIC", "IlluminaHumanMethylation27k"),
  mn = NULL,
  un = NULL,
  bn = NULL,
  perc = 1,
  compositeCellType = "Blood",
  probeSelect = "auto",
  cellTypes = c("CD8T", "CD4T", "NK", "Bcell", "Mono", "Gran"),
)"
```r
returnAll = FALSE,
meanPlot = FALSE,
verbose=TRUE,
...)
```

**Arguments**

- **object**: An object of class methylumiset, which contains (un)normalised methylated and unmethylated intensities.
- **mn**: if NULL will call methylated(object), otherwise can be given matrix of identical dimensions to object.
- **un**: if NULL will call unmethylated(object), otherwise can be given matrix of identical dimensions to object.
- **bn**: if NULL will call betas(object), otherwise can be given matrix of identical dimensions to object.
- **perc**: Percentage of query-samples to use to normalise reference dataset. This should be 1 unless using a very large data-set then lowering this will allow for an increase in performance.
- **compositeCellType**: Which composite cell type is being deconvoluted. Should be either "Blood", "CordBlood", or "DLPFC".
- **probeSelect**: How should probes be selected to distinguish cell types? Options include "both", which selects an equal number (50) of probes (with F-stat p-value < 1E-8) with the greatest magnitude of effect from the hyper- and hypo-methylated sides, and "any", which selects the 100 probes (with F-stat p-value < 1E-8) with the greatest magnitude of difference regardless of direction of effect. Default input "auto" will use "any" for cord blood and "both" otherwise, in line with previous versions of this function and/or our recommendations. Please see the references for more details.
- **cellTypes**: Which cell types, from the reference object, should be used for the deconvolution? See details.
- **referencePlatform**: The platform for the reference dataset; if the input rgSet belongs to another platform, it will be converted using `convertArray`.
- **returnAll**: Should the composition table and the normalized user supplied data be return?
- **verbose**: Should the function be verbose?
- **meanPlot**: Whether to plots the average DNA methylation across the cell-type discriminating probes within the mixed and sorted samples.
- **...**: Other arguments, i.e arguments passed to plots.

**Details**

See `estimateCellCounts` for more information regarding the exact details. `estimateCellCounts.wmln` differs slightly, as it will impose the quantiles of type I and II probes onto the reference Dataset rather than normalising the two together. This is 1) More memory efficient and 2) Faster - due to
not having to normalise out a very small effect the other 60 samples from the reference set will have on the remaining quantiles.

Optionally, a proportion of samples can be used to derive quantiles when there are more than 1000 samples in a dataset, this will further increase performance of the code at a cost of precision. If data is pre-normalised a minimum of two samples are required.

---

**estimateSex**

*Predict sex by using robust sex-related CpG sites on ChrX and ChrY*

**Description**

Predict sex by using robust sex-related CpG sites on ChrX and ChrY

**Usage**

```r
estimateSex(betas, do_plot = FALSE)
```

**Arguments**

- `betas`: A matrix with sample IDs as column names, and probe names as row names, ideally: \( \text{beta} = \frac{M}{(M + U + 100)} \). Take a look at an example betas with: 
  `data(melon); print(betas(melon)[1:10, 1:3])`.

- `do_plot`: logical. Should plot the predicted results? Default: FALSE

**Value**

dataframe contains predicted sex information.

**Author(s)**


**Examples**

```r
data(melon)
pred_XY <- estimateSex(betas(melon), do_plot=TRUE)
```
SNP derived performance metrics for Illumina 450K DNA methylation arrays.

Description
A very simple genotype calling by one-dimensional K-means clustering is performed on each SNP, and for those SNPs where there are three genotypes, the squared deviations are summed for each genotype (similar to a standard deviation for each of allele A homozygote, heterozygote and allele B homozygote). By default these are further divided by the square root of the number of samples to get a standard error-like statistic.

Usage
```
genki(bn, g = getsnp(rownames(bn)), se = TRUE)
```

Arguments
- **bn**: a matrix of beta values (default method), a MethyLumiSet object (methylumi package), a MethylSet or RGChannelSet object (minfi package) or a exprmethy450 object (IMA package).
- **g**: vector of SNP names
- **se**: TRUE or FALSE specifies whether to calculate the standard error-like statistic

Details
There are 65 well-behaved SNP genotyping probes included on the array. These each produce a distribution of betas with tight peaks for the three possible genotypes, which will be broadened by technical variation between samples. The spread of the peaks is thus usable as a performance metric.

Value
a vector of 3 values for the dispersion of the three genotype peaks (AA, AB, BB : low, medium and high beta values)

Note
Corrected RGChannelSet methods - 12/10/2015

Author(s)
Leonard.Schalkwyk@kcl.ac.uk

References
**Examples**

```r
#MethyLumiSet method
data(melon)
genki(melon)

#MethyLumiSet method after normalization
melon.dasen <- dasen(melon)
genki(melon.dasen)
```

---

### Description

Methods for function `genki` in package `wateRmelon`. Please see `genki` for details of the calculation of this standard-error performance metric.

### Methods:

```r
signature(betas = "exprmethy450") all of the methods simply extract betas from the data object (which can be a exprmethy450, MethylSet, MethyLumiSe, or RGChannelSet) and calculate the metric.
```

---

### genkme

*Internal functions for genotype-based normalization metrics*

### Description

- `genkme` - genotype calling with 1d k-means
- `genkus` - apply genkme to available SNPs
- `getsnp` - grep the rs-numbered probes
- `gcose` - calculate between-sample SNP standard error
- `gcoms` - calculate between-sample SNP mean-squared deviation

### Usage

```r
genkme(y, peaks = c(0.2, 0.5, 0.8))
```

### Arguments

- **y** a vector or matrix of numeric values (betas, between 0 and 1)
- **peaks** initial values for cluster positions

---
Details
- see genki

Value
- see genki

Author(s)
Leonard.Schalkwyk@kcl.ac.uk

References

| got                              | Internal functions for Illumina i450 normalization functions |

Description
- got and fot find the annotation column differentiating type I and type II assays in MethylSet (got) or MethyLumiSet (fot) objects. pop extracts columns from IlluminaHumanMethylation450k.db

Usage
- got(obj)
- fot(x)

Arguments
- x a MethyLumiSet
- obj a MethylSet

Details
- got returns a character vector of 'I' and 'II', fot returns the index of the relevant column. pop returns a data frame

Author(s)
Leonard.Schalkwyk@kcl.ac.uk

References
**idet**

*idet - identify idats by a hash of the addresses*

**Description**

`idet` - identify idats by a hash of the addresses

**Usage**

`idet(idat)`

**Arguments**

`idat` can be an idat file or the list produced by reading one with `readIDAT()`

**Details**

This function is a response to the fact that IlluminaHumanMethylationEPIC and EPICv2 idats both have the ChipType "BeadChip 8x5" but different manifests. They did have different numbers of addresses. Subsequently we have had some confusion. This hash (certainly taken together with the ChipType) should be bombproof as an identifier. Warning: this is slow.

**Value**

three strings: ChipType, an md5 hash of the MidBlock (address vector) and if known the annotation name

---

**iDMR**

*Imprinting differentially methylated region probes of Illumina 450 arrays*

**Description**

A character vector of 227 propes on the Illumina 450k methylation array

**Usage**

`data(iDMR)`

**Format**

The format is: chr [1:227] "cg00000029" "cg00155882" "cg00576435" "cg00702231" "cg00765653" "cg00766368" ...

**Source**

DMR coordinates from https://atlas.genetics.kcl.ac.uk/
References


Examples

data(iDMR)
## maybe str(iDMR); plot(iDMR) ...

melon Small MethyLumi data set for examples and testing

Description

This object was derived using methylumiR on an edited GenomeStudio file containing a small subset of features. It works with all of the wateRmelon package beta functions (see dasen and metrics (see genki, seabi, and dmrse_col) except for swan.

Usage

data(melon)

Format

MethyLumiSet with assayData containing 3363 features, 12 samples

Source


Examples

library(methylumi)
data(melon)
boxplot(log(methylated(melon)), las=2)
## maybe str(melon); plot(melon) ...
metrics

Calculate a full set of 450K normalization/performance metrics

Description

Calculate X-chromosome, SNP and imprinting DMR metrics for a matrix of betas from an Illumina 450K Human DNA methylation array. Requires precalculated t-test p-values for sex differences, a list of X-chromosome features and of imprinting DMR features.

Usage

metrics(betas, pv, X, idmr = iDMR, subset = NULL)

Arguments

betas a matrix of betas, each row representing a probe, each column a sample
pv a vector of p-values such as produced by sextest, one per row of betas
X a logical vector of the same length as pv, indicating whether each probe is mapped to the X-chromosome
idmr a character vector of probe names known to be in imprinting DMRs. Can be obtained with iDMR() or data(idMR)
subset index or character vector giving a subset of betas to be tested

Value

dmrse_row see dmrse_row
dmrse_col see dmrse_col
dmrse see dmrse
gcoms_a see genki
gcope_a see genki
gcoms_b see genki
gcope_b see genki
gcoms_c see genki
gcope_c see genki
seabird see seabi

Author(s)

Leonard.Schalkwyk@kcl.ac.uk

References

Examples

data(melon)
melon.dasen <- dasen(melon)
bn <- betas(melon.dasen)
X <- melon.dasen@featureData@data$CHR=='X'
data(iDMR)
sex <- pData(melon.dasen)$sex
pv <- sextest(bn, sex)
melon.metrics <- metrics(bn, pv, X, idmr = iDMR, subset = NULL)

outlyx

Identify Outliers within Methylumi and Minfi packaged objects

Description

Seeks to identify outliers based on multiple (currently 2) outlier detection methods for methylumi and minfi packaged objects.

Usage

outlyx(x, iqr=TRUE, iqrP=2, pc=1,
      mv=TRUE, mvP=0.15, plot=TRUE, ...)

Arguments

x A MethylLumiSet, MethylSet, RGChannelSet object or matrix containing raw
    betas.

iqr If TRUE, the outliers based on interquartile ranges will be determined

iqrP The number of interquartile ranges outliers are to be identified from designated
      principle component.

pc Desired principal component for outlier identification - only used if other prin-
      cipal components want to be discriminated, only used for IQR outlier detection.

mv If TRUE, the outliers will detected using pcout

mvP Arbitrary cut-off point for identifying outliers via pcout

plot If TRUE, alongside regular output, a plot will be constructed displaying relative
      'location' of each sample. Outliers are those that fall within the highlighted
      regions.

... Additional arguments passed to pcout

Value

Returns a dataframe of TRUE/FALSE per sample where TRUE is outlying. Dataframe contains 3
columns, the first column (iqr) denotes samples which are outlying according to IQR on Principal
component 1, the second column (mv) denotes outliers according to mahalanobis distances. And
the third column (outliers) denotes samples that are TRUE in the first two columns.
Note

May perform poorly on normalized data

Author(s)

Tyler Gorrie-Stone - tgorri@essex.ac.uk

Examples

```r
library(wateRmelon)
data(melon)
outliers <- outlyx(melon, iqr=TRUE, iqrP=2, pc=1,
                    mv=TRUE, mvP=0.15, plot=TRUE)
```

Description

Methods for function outlyx, please see outlyx for details of how function performs.

Methods

```r
signature(x = "MethyLumiSet") all of the methods simply extract betas from the data object
                      (which can be a MethyLumiSet, MethyLumiSet, or RGChannelSet) and calculates the outliers.
```

pfilter

Basic data filtering for Illumina 450 methylation data

Description

The pfilter function filters data sets based on bead count and detection p-values. The user can set their own thresholds or use the default pfilter settings. pfilter will take data matrices of beta values, signal intensities and annotation data, but will also take methylumi (MethyLumiSet) or minfi (RGChannelSetExtended) objects. However it has come to our attention that data read in using the various packages and input methods will give subtly variable data output as they calculate detection p-value and beta values differently, and do/don’t give information about beadcount. The pfilter function does not correct for this, but simply uses the detection p-value and bead count provided by each package.

Usage

```r
pfilter(mn, un, bn, da, pn, bc, perCount=NULL, pthresh = NULL, perc = NULL,
        pthresh = NULL, logical.return=FALSE)
```
Arguments

\( mn \) matrix of methylated signal intensities, each column representing a sample (default method), or an object for which a method is available e.g MethyLumiSet or RGChannelSetExtended. N.B. Bead count filtering will not work unless data read in as an RGChannelSetExtended rather than an RGChannelSet.

\( un \) matrix of unmethylated signal intensities, each column representing a sample (default method) or NULL when \( mn \) is a MethyLumiSet or RGChannelSetExtended object

\( bn \) matrix of precalculated betas, each column representing a sample, or NULL when \( mn \) is a MethyLumiSet or RGChannelSetExtended object

\( da \) annotation data frame, such as \( x@featureData@data \) #methyumi package, or NULL when \( mn \) is a MethyLumiSet or RGChannelSetExtended object

\( pn \) matrix of detection p-values, each column representing a sample, a MethyLumiSet or RGChannelSetExtended object

\( bc \) matrix of arbitrary values, each column representing a sample and each row representing a probe, in which "NA" represents beadcount <3, or NULL when \( mn \) is a MethyLumiSet or RGChannelSetExtended object

\( perCount \) remove sites having this percentage of samples with a beadcount <3, default set to 5

\( pnthresh \) cutoff for detection p-value, default set to 0.05

\( perc \) remove samples having this percentage of sites with a detection p-value greater than \( pnthresh \), default set to 1

\( pthresh \) remove sites having this percentage of samples with a detection p-value greater than \( pnthresh \), default set to 1

\( logical.return \) If it is TRUE, FALSE or TRUE is returned to indicate success

Value

a filtered MethyLumiSet or
a list of the filtered matrices:
\( mn \) : methylated intensities
\( un \) : unmethylated intensities
\( bn \) : betas
\( da \) : feature data
or
a filtered MethylSet object.

Methods

\texttt{signature(mn = "MethyLumiSet")} This is used for performing the \texttt{pfilter} method on MethyLumiSet objects produced by methyumiR.

\texttt{signature(mn = "RGChannelSetExtended")} This is used for performing the \texttt{pfilter} method on RGChannelSetExtended objects produced by minfi.
Note

Adjusted RGChannelSetExtended methods - 12/10/2015 Now outputs a MethylSet object using preprocessRaw from minfi.

Author(s)

Ruth.Pidsley@kcl.ac.uk

References


Examples

# MethyLumiSet method
data(melon)
melon.pf <- pfilter(melon)

pwod

---

**Probes Outlier Detection**

Description

'P'robe-'W'ise 'O'utlier 'D'etection via interquartile ranges.

Usage

pwod(object, mul=4)

Arguments

  object  MethyLumiSet, RGChannelSet, MethylSet object or matrix containing betas.
  mul     Number of interquartile ranges used to determine outlying probes. Default is 4 to ensure only very obvious outliers are removed.

Details

Detects outlying probes across arrays in methylumi and minfi objects. Outliers are probable low MAF/SNP heterozygotes.

Value

Returns supplied beta matrix with outlying probes coerced to NA

Author(s)

Tyler Gorrie-Stone - tgorri@essex.ac.uk
Examples

```r
library(wateRmelon)
data(melon)
cattle <- betas(melon)
new.betas <- pwod(cattle, mul=4)
```

Description

Methods for function `pwod`, please see `pwod` for details of how function performs.

Methods

signature(object = "MethyLumiSet") all of the methods simply extract betas from the data object (which can be a MethylSet, MethyLumiSet, or RGChannelSet) and calculates the outliers.

qual

A measure of Normalization Violence

Description

Calculates 4 metrics to assess the degree of difference between normalized and raw betas.

Usage

```r
qual(norm, raw)
```

Arguments

- `norm`: Matrix of normalized betas
- `raw`: Matrix of raw betas

Value

Returns data.frame containing rmsd, sdd, sadd and srms for each sample (columns) in supplied matrices.

Author(s)

Leo Schalkwyk
Examples

```r
library(wateRmelon)
data(melon)
d.melon <- dasen(melon)
raw.bet <- betas(melon)
norm.bet <- betas(d.melon)
qual(norm=norm.bet, raw=raw.bet)
```

Description

`read.manifest` - read in csv format Illumina chip manifest files

Usage

```r
read.manifest(file)
```

Arguments

- `file`

Details

This function is probably not much use for calling directly, it mostly exists to be called by canno.

Value

A list of dataframes of data prepared for making `IlluminaMethylationManifest`

Description

`readEPIC` - reads Epic arrays from raw idats into `MethyLumiSet` objects from directory.

Usage

```r
readEPIC(idatPath, barcodes=NULL, pdat=NULL, parallel=F, n=T, oob=F, force=F, ...)
```
Arguments

idatPath  Path directory where .idat files are located. readEPIC looks in the specified path and converts all .idats within path to relevant barcodes, which is then passed to a modified version of methylumIDAT to parse all idats present in the specified directory.

barcodes  If NULL, function will search supplied argument in "idatPath" for all idats within directory. If given a vector of barcodes, readEPIC will search for those specific barcodes within the idatPath supplied.

parallel  If TRUE, an attempt will be made to process using multiple cores on a multicore machine.

pdat  A data.frame describing the samples. A special column named "barcodes" can be used to specify the barcodes to be read when using methylumIDATepic. See methylumIDAT for usage

n  If TRUE, beadcounts from .idats will be included in final object

oob  If TRUE, out-of-band (OOB) or opposite-channel signals will be kept

force  If TRUE, will combine EPIC IDATs read with differing dmaps

...  Additional arguments passed to methylumIDAT

Details

Read a set of .idat files within a file directory and return a MethylumiSet object.

Value

A MethylumiSet object.

Note

Contains heavily modified version of methylumIDAT and other accessory functions used to construct a MethylumiSet object, specifically tailored for EPIC arrays. readEPIC can also handle 450k and 27k arrays as methylumIDAT functionality for these platforms remains unchanged.

Alternatively it is possible to invoke methylumIDATepic to use the modified version methylumIDAT, which has similar usage.

EPIC manifest has since been updated to B4 version, which has notably fewer probes than previous manifests. It is entirely possible that we will migrate to the manifest packages available on BioConductor and allow for versioning control.

Author(s)

Tyler Gorrie-Stone - tgorri@essex.ac.uk

References

methylum
readPepo

Examples

```r
# Fictitious file pathway
# path <- "Data/Experiment/Idatlocation"
# data <- readEPIC(path, barcodes = NULL oob=F, n=T)
```

Description

readPepo - (unfinished) read any kind of Illumina DNA methylation array idat files into a methylumi object

Usage

```r
readPepo(
  idatdir = ".",
  filelist = NULL,
  barcodelist = NULL,
  manifest = NULL,
  parallel = F,
  n = F,
  pdat = NULL,
  oob = F,
  two = TRUE
)
```

Arguments

- **idatdir**: the directory with the idat files. Currently only handle one directory.
- **filelist**: optional list of idat files to process.
- **barcodelist**: optional list of barcodes to process.
- **manifest**: name of a IlluminaMethylationManifest object or a csv format manifest. If missing, will run idet() on one of the idat files.
- **parallel**: try to use multiple cores.
- **n**: keep beadcounts.
- **pdat**: optional data.frame describing the samples.
- **two**: are there two different assay types (true of human methylation arrays except 27k)
- **keep**: out-of-band (OOB) or opposite-channel signals

Value

A ‘MethyLumiSet’ object.
Calculate a performance metric based on male-female differences for Illumina methylation 450K arrays

Description

Calculates an area under ROC curve - based metric for Illumina 450K data using a t-test for male-female difference as the predictor for X-chromosome location of probes. The metric is 1-area so that small values indicate good performance, to match our other, standard error based metrics gcose and dmrse. Note that this requires both male and female samples of known sex and can be slow to compute due to running a t-test on every probe.

Usage

seabi(bn, stop = 1, sex, X)

Arguments

bn a matrix of betas (default method) or an object containing betas i.e. a MethylumiSet object (methylumi package), a MethylSet or RGChannelSet object (minfi package) or a exprmethy450 object (IMA package).

stop partial area under curve is calculated if stop value <1 is provided

sex a factor giving the sex of each sample (column)

X a logical vector of length equal to the number of probes, true for features mapped to X-chromosome

Value

a value between 0 and 1. values close to zero indicate high data quality as judged by the ability to discriminate male from female X-chromosome DNA methylation.

Author(s)

leonard.schalkwyk@kcl.ac.uk

References


Examples

library(methylumi)
data(melon)
sex <- pData(melon)$sex
X <- melon@featureData@data$CHR=='X'
seabi(betas(melon), sex=sex, X=X)
# methylumi method
seabi(melon, sex=sex, X=X)

## seabi-methods

### Methods for Function seabi in Package wateRmelon

#### Description

Methods for function `seabi` in package `wateRmelon`. Please see `seabi` for details of the calculation of this ROC AUC performance metric.

#### Methods

- `signature(betas = "exprmethy450")` all of the methods simply extract betas from the data object (which can be a `exprmethy450`, `MethylSet`, `MethylumiSe`, or `RGChannelSet`) and calculate the metric. All the methods also require a factor differentiating male from female samples.

## seabird

#### Calculate ROC area-under-curve for X-chromosome sex differences (internal function for calculating the seabi metric)

#### Description

This is a wrapper for the prediction and performance functions from the ROCR package that takes a vector of p-values and a vector of true or false for being on the X. See `seabi` function which does everything.

#### Usage

```
seabird(pr, stop = 1, X)
```

#### Arguments

- `pr` a vector of p-values, such as calculated by `seabird`
- `stop` fraction for partial area under curve. For example 0.1 gives you the area for the lowest 10% of p-values.
- `X` logical vector the same length as `pr`, true for features mapped to X-chromosome

#### Value

Returns an area value between 0 and 1, where 1 is the best possible performance.

#### Author(s)

Leonard C Schalkwyk 2012 Leonard.Schalkwyk@kcl.ac.uk
References


Description

This is a wrapper for t.test which does the equivalent of a Student t-test for difference in betas between males and females for each row of a matrix of betas.

Usage

sextest(betas, sex, ...)

Arguments

betas a matrix of betas, each row is a probe, each column a sample
sex a factor with 2 levels for male and female
... additional arguments to be passed to t.test

Value

Returns a vector of p-values of length equal to the number of rows of betas

Author(s)

Leonard.Schalkwyk@kcl.ac.uk

References


See Also

seabi seabird
**smokp**

### Examples

```r
#MethyLumiSet method
data(melon)
sex <- pData(melon)$sex
melon.sextest <- sextest(betas(melon), sex)

#MethyLumiSet method with quality control step
data(melon)
melon.dasen <- dasen(melon)
sex <- pData(melon.dasen)$sex
melon.sextest <- sextest(betas(melon.dasen), sex)
```

---

**smokp**

*Smoking Prediction from methylomic expression data*

### Description

Predict smoking from samples using various methods

### Usage

```r
smokp(betas, method, sst)
```

### Arguments

- **betas**: Matrix of betas or MethyLumiSet or MethylSet object. Rows are Illumina IDs referring to CpG sites and Columns refer to samples or participants.
- **method**: Currently: 'AHRR', 'McCartney', 'Maas', 'Sugden', 'Teschendorff', 'Yu', 'Gao', 'Yang', 'Zhang', 'Wen', 'Langdon', 'SSSt', 'Packyears', 'Cessation', and 'All'. If "All" smokp will seek to predict smoking using all methods else will use the method specified. Default is "SSSt". If "Teschendorff", "Yu", "Gao", "Yang" or "Langdon" specified then smoking status is required.
- **sst**: Named vector describing smoking status, coded as 'Current', 'Former', or 'Never', of participant for each sample (where names match rownames of betas).

### Value

Returns data frame of predicted smoking per sample.

### Author(s)

- Original Functions: See References.
- wateRmelon: Tyler Gorrie-Stone, Leo Schalkwyk, Louis El Khoury
- smokp: Alexandria Andrayas
References


Examples

```r
data(melon)
# note, melon is not a complete dataset, does not work with all methods
smokp(melon, method="McCartney", sst=NULL)
```

wm_internal

Internal functions for readEPIC and other watermelon functions introduced in v 1.13.1

Description

few if any functions of interest to users

Usage

```r
DataToNChannelSet2(mats, chans = c(Cy3 = "GRN", Cy5 = "RED"), parallel = F, protocol.data = F, IDAT = TRUE)
```

Arguments

- `mats`
- `chans`
- `parallel`
- `protocol.data`
- `IDAT`
- `force`
Index

* Bisulphite Conversion Rate
  bscon, 15
* MethyLumiSet
  melon, 34
* QC data
  bscon, 15
* datasets
  iDMR, 33
  melon, 34
* methods
  as.methylumi-methods, 9
  colnames-methods, 17
  dmrse-methods, 26
  genki-methods, 31
  outlyx-methods, 37
  pwod-methods, 40
  seabi-methods, 45
* outlier
  outlyx, 36
* package
  wateRmelon-package, 3
  .createAnnotation, 4
  .getManifestString, 4
  adaptRefQuantiles, 5
  adjustedDassen, 6
  adjustedFunnorm, 7
  age_coefficients (agep), 8
  agep, 8
  agep,MethylSet-method (agep), 8
  agep,MethyLumiSet-method (agep), 8
  agep,RGChannelSet-method (agep), 8
  anSNP (wm_internal), 49
  anti.trafo (wm_internal), 49
  aoget (wm_internal), 49
  as.methylumi, 18, 20
  as.methylumi (as.methylumi-methods), 9
  as.methylumi,ANY-method (as.methylumi-methods), 9
  as.methylumi,MethylSet-method (as.methylumi-methods), 9
  as.methylumi,MethyLumiSet-method (as.methylumi-methods), 9
  as.methylumi-methods, 9
  auc_probability (wm_internal), 49
  beadc, 10
  beadcount, 11
  Beta2M, 12
  betaqn (dasen), 18
  betaqn,exprmethyl450-method (betaqn-exprmethyl450-methods), 12
  betaqn,MethylSet-method (dasen-minfi-methods), 22
  betaqn,MethyLumiSet-method (dasen-methods), 21
  betaqn,RGChannelSet-method (dasen-minfi-methods), 22
  betaqn-exprmethyl450-methods, 12
  bfp (wm_internal), 49
  bgIntensitySwan.methylumi (adaptRefQuantiles), 5
  BMIQ, 13
  BMIQ,ANY-method (BMIQ), 13
  BMIQ,MethylSet-method (BMIQ), 13
  BMIQ,MethyLumiSet-method (BMIQ), 13
  BMIQ-methods (BMIQ), 13
  bscon, 15
  bscon,MethylSet-method (bscon), 15
  bscon,RGChannelSet-method (bscon), 15
  bscon_methy (wm_internal), 49
  bscon_minfi (wm_internal), 49
  canno, 16
  CheckBMIQ (BMIQ), 13
  coef (agep), 8
  colnames,MethyLumiSet-method (colnames-methods), 17
colnames-methods, 17
columnMatrix (wm_internal), 49
combo, 17
concatenateMatrices (adaptRefQuantiles), 5
convertArray, 28
coRankedMatrices (adaptRefQuantiles), 5
correctI (Beta2M), 12
correctII (Beta2M), 12
danen (dasen), 18
danen, MethylSet-method (dasen-minfi-methods), 22
danen, MethyLumiSet-method (dasen-methods), 21
danen, RGChannelSet-method (dasen-minfi-methods), 22
danes (dasen), 18
danes, MethylSet-method (dasen-minfi-methods), 22
danes, MethyLumiSet-method (dasen-methods), 21
danes, RGChannelSet-method (dasen-minfi-methods), 22
danet (dasen), 18
danet, MethylSet-method (dasen-minfi-methods), 22
danet, MethyLumiSet-method (dasen-methods), 21
danet, RGChannelSet-method (dasen-minfi-methods), 22
dasen, 18, 34
dasen, MethylSet-method (dasen-minfi-methods), 22
dasen, MethyLumiSet-method (dasen-methods), 21
dasen, RGChannelSet-method (dasen-minfi-methods), 22
dasen-methods, 21
dasen-minfi-methods, 22
dataDetectPval2NA (adaptRefQuantiles), 5
DataToNChannelSet2 (wm_internal), 49
daten1 (dasen), 18
daten1, MethylSet-method (dasen-minfi-methods), 22
daten1, MethyLumiSet-method (dasen-methods), 21
daten1, RGChannelSet-method (dasen-minfi-methods), 22
daten2 (dasen), 18
daten2, MethylSet-method (dasen-minfi-methods), 22
daten2, MethyLumiSet-method (dasen-methods), 21
daten2, RGChannelSet-method (dasen-minfi-methods), 22
db1, 24
designIItoMandU2 (wm_internal), 49
designItoMandU2 (wm_internal), 49
detectionPval.filter (adaptRefQuantiles), 5
dfort (wm_internal), 49
dfs2 (db1), 24
dfsfit (db1), 24
dmrse, 25, 26, 44
dmrse, exprmethy450-method (dmrse-methods), 26
dmrse, MethylSet-method (dmrse-methods), 26
dmrse, MethyLumiSet-method (dmrse-methods), 26
dmrse, RGChannelSet-method (dmrse-methods), 26
dmrse-methods, 26
dmrse_col, 34
dmrse_col (dmrse), 25
dmrse_col, exprmethy450-method (dmrse-methods), 26
dmrse_col, MethylSet-method (dmrse-methods), 26
dmrse_col, MethyLumiSet-method (dmrse-methods), 26
dmrse_col, RGChannelSet-method (dmrse-methods), 26
dmrse_col-methods (dmrse-methods), 26
dmrse_row (dmrse), 25
dmrse_row, exprmethy450-method (dmrse-methods), 26
dmrse_row, MethylSet-method (dmrse-methods), 26
dmrse_row, MethyLumiSet-method (dmrse-methods), 26
dmrse_row, RGChannelSet-method (dmrse-methods), 26
dmrse_row-methods (dmrse-methods), 26
epic.controls (readEPIC), 41
epicv2clean (epicv2clean.default), 27
epicv2clean.default, 27
estimateCellCounts, 27, 27, 28
estimateSex, 29
extractAssayDataFromList2
  (wm_internal), 49
filterXY (adaptRefQuantiles), 5
findAnnotationProbes
  (adaptRefQuantiles), 5
got (.getManifestString), 4
got (got), 32
fuks (dasen), 18
fuks, exprmethy450-method
  (betaqn-exrmethy450-methods), 12
fuks, MethylSet-method
  (dasen-minfi-methods), 22
fuks, MethyLumiSet-method
  (dasen-methods), 21
fuks, RGChannelSet-method
  (dasen-minfi-methods), 22
gcoms (genkme), 31
gcose, 44
gcose (genkme), 31
genall (wm_internal), 49
generateManifest (wm_internal), 49
genki, 26, 30, 31, 32, 34
genki, exprmethy450-method
  (genki-methods), 31
genki, MethylSet-method (genki-methods), 31
genki, MethyLumiSet-method
  (genki-methods), 31
genki, RGChannelSet-method
  (genki-methods), 31
genki-methods, 31
genkme, 31
genkus (genkme), 31
genme (wm_internal), 49
genus (wm_internal), 49
getColumns (as.methylumi-methods), 9
getControlProbes2 (wm_internal), 49
getMethylationBeadMappers2
  (wm_internal), 49
goodSNP (wm_internal), 49
got, 32
got (.getManifestString), 4
hannumCoef (agep), 8
IDATsToMatrices2 (wm_internal), 49
IDAToMatrix2 (wm_internal), 49
idet, 33
iDMR, 33
iqrFun (wm_internal), 49
loadMethylumi2 (adaptRefQuantiles), 5
lumiMethyR2 (adaptRefQuantiles), 5
M2Beta (Beta2M), 12
melon, 34
mergeProbeDesigns2 (wm_internal), 49
methylumiIDATepic (wm_internal), 49
methylumiR, 34
metrics, 35
mvFun (wm_internal), 49
nanes (dasen), 18
nanes, MethylSet-method
  (dasen-minfi-methods), 22
nanes, MethyLumiSet-method
  (dasen-methods), 21
nanes, RGChannelSet-method
  (dasen-minfi-methods), 22
nanet (dasen), 18
nanet, MethylSet-method
  (dasen-minfi-methods), 22
nanet, MethyLumiSet-method
  (dasen-methods), 21
nanet, RGChannelSet-method
  (dasen-minfi-methods), 22
nasen (dasen), 18
nasen, MethylSet-method
  (dasen-minfi-methods), 22
nasen, MethyLumiSet-method
  (dasen-methods), 21
nasen, RGChannelSet-method
  (dasen-minfi-methods), 22
naten (dasen), 18
naten, MethylSet-method
  (dasen-minfi-methods), 22
naten, MethyLumiSet-method
  (dasen-methods), 21
naten, RGChannelSet-method
  (dasen-minfi-methods), 22
naten, RGChannelSet-method
  (dasen-minfi-methods), 22
pbBeadsFilter (adaptRefQuantiles), 5
NChannelSetToMethyLumiSet2
  (wm_internal), 49
normalize.quantiles2
  (adaptRefQuantiles), 5
normalizeIlluminaMethylation
  (adaptRefQuantiles), 5
outlyx, 36
outlyx, MethylSet-method
  (outlyx-methods), 37
outlyx, MethyLumiSet-method
  (outlyx-methods), 37
outlyx, RGChannelSet-method
  (outlyx-methods), 37
outlyx-methods, 37
oxyscale (wm_internal), 49
p_dfsfit (wm_internal), 49
pcouted (wm_internal), 49
pfilter, 20, 37
pfilter, MethyLumiSet-method (pfilter), 37
pfilter, RGChannelSetExtended-method (pfilter), 37
pfilter-methods (pfilter), 37
pipelineIlluminaMethylation.batch
  (adaptRefQuantiles), 5
plot.predicted.sex (wm_internal), 49
pop (.getManifestString), 4
preprocessIlluminaMethylation
  (adaptRefQuantiles), 5
pwod, 39
pwod, MethylSet-method (pwod-methods), 40
pwod, MethyLumiSet-method
  (pwod-methods), 40
pwod, RGChannelSet-method
  (pwod-methods), 40
pwod-methods, 40
qual, 40
read.manifest, 41
readEPIC, 41
readPepo, 43
referenceQuantiles (adaptRefQuantiles), 5
robustQuantileNorm_Illumina450K
  (adaptRefQuantiles), 5
seabi, 26, 34, 44, 45, 46
seabi, exprmeth450-method
  (seabi-methods), 45
seabi, MethylSet-method (seabi-methods), 45
seabi, MethyLumiSet-method
  (seabi-methods), 45
seabi, RGChannelSet-method
  (seabi-methods), 45
seabi-methods, 45
seabird, 45, 46
seabird2 (wm_internal), 49
sextest, 46
smokp, 47
smokp, MethylSet-method (smokp), 47
smokp, MethyLumiSet-method (smokp), 47
smokp, RGChannelSet-method (smokp), 47
smokp_cpgs (smokp), 47
sort_order (wm_internal), 49
subbo (wm_internal), 49
summits (Beta2M), 12
swan, 10, 34
swan (dasen), 18
swan, MethylSet-method
  (dasen-minfi-methods), 22
swan, MethyLumiSet-method
  (dasen-methods), 21
swan, RGChannelSet-method
  (dasen-minfi-methods), 22
tie_norm (wm_internal), 49
tost, 5
tost (dasen), 18
tost, MethylSet-method
  (dasen-minfi-methods), 22
tost, MethyLumiSet-method
  (dasen-methods), 21
tost, RGChannelSet-method
  (dasen-minfi-methods), 22
trafo (wm_internal), 49
uniqueAnnotationCategory
  (adaptRefQuantiles), 5
uSexQN (wm_internal), 49
uSexQN, MethySet-method (wm_internal), 49
uSexQN, MethylumiSet-method (wm_internal), 49
uSexQN, RGChannelSet-method (wm_internal), 49
uSexQNeengine (wm_internal), 49

watermelon (watermelon-package), 3
watermelon-package, 3
wm_internal, 49