Package ‘wateRmelon’

March 29, 2017

Type Package
Title Illumina 450 methylation array normalization and metrics
Version 1.18.0
Date 2016-09-14
Author Leonard C Schalkwyk, Ruth Pidsley, Chloe CY Wong, with functions contributed by Nizar Touleimat, Matthieu Defrance, Andrew Teschendorff, Jovana Maksimovic, Tyler Gorrie-Stone, Louis El Khoury
Maintainer Leo <lschal@essex.ac.uk>
Description 15 flavors of betas and three performance metrics, with methods for objects produced by methylumi and minfi packages.
License GPL-3
Depends R (>= 2.10), Biobase, limma, methods, matrixStats, methylumi, lumi, ROC, IlluminaHumanMethylation450kanno.ilmn12.hg19, illuminaio
Imports Biobase
Enhances minfi
Suggests RPMM
LazyLoad yes
biocViews DNA Methylation, Microarray, TwoChannel, Preprocessing, Quality Control

1
adaptRefQuantiles

watermelon-package  **Illumina 450K arrays: normalization and performance metrics**

**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**


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`

**Value**

see `tost`

**Author(s)**

Nizar Touleimat, wrapper by Leonard.Schalkwy1@kcl.ac.uk
References


agep

Description

Predict age of samples using Horvath’s Coefficients

Usage

agep(betas, coeff=NULL, verbose=FALSE,...)

Arguments

betas Matrix of betas or MethyLumiSet or MethylSet object.
coeff If NULL, will default to Horvath’s coefficients. Otherwise can be supplied with vector of own coefficients with an intercept as first element.
verbose If TRUE, agep will additionally print out names of probes per sample that are missing from coefficient list within supplied beta matrix.
... To pass to arguments to downstream functions to specify adult.age

Value

Returns matrix of predicted ages per sample.

Author(s)

Original Functions: Steve Horvath
wateRmelon Implementation: 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)
# or
agep(betas(melon), coeff=NULL)
as.methylumi-methods

Methods for Function as.methylumi

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, fd = c("CHR", "DESIGN"), ad=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

da annotation data frame, such as x@featureData@data #methylumi package. If NULL (the default), the IlluminaHumanMethylation450kmanifest package is used. See the fd argument

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

fd vector of items of featureData, which by default is just the chromosome and DESIGN (ie typeI or type II assay). Other data can be included using the fd argument, available data is listed by the function getColumns()

ad 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 typeI 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 methylumiIDAT 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 = NULL, ad=NULL)
beadc

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

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 beadc 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

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
Beta2M

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


| Beta2M |
|-----------------
| Internal functions for peak.correction (fuks) |

Description
Internal functions for peak.correction

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

Description
Quantile normalize betas from exprmethy450 objects

Arguments
bn An exprmethy450 object.
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.

methods
betaqn( bn ) fuks( 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)
```

## S4 method for signature 'VarMethyLumiSet'

```r
BMIQ(beta.v, nL=3, doH=TRUE, nfit=5000, th1.v=c(0.2,0.75), th2.v=NULL, niter=5, tol=0.001, plots=FALSE, CheckBMIQ(beta.v, design.v, pnbeta.v)
```

**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 buest 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 buest 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

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**  
*Calculate bisulphite conversion*

**Description**

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

**Usage**

```r
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 bisulphite 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
```
Methods for Function `colnames` in Package `wateRmelon`

**Description**

Methods for function `colnames` in package `wateRmelon`.

**Methods**

- `signature(x = "MethyLumiSet")` returns the sample names

---

**combo**

*Combine MethyLumiSet objects*

**Description**

This is a wrapper for combining different MethyLumiSet objects.

**Usage**

`combo(...)`

**Arguments**

- `...` Eventually, any number of MethyLumiSet objects. Currently only guaranteed for 2 objects.

**Details**

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

**Value**

A `MethyLumiSet`. The `assayData`, `QCdata`, `experimentData`, `protocolData` and `phenoData` are joined on `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 `make.names()` first.

**Author(s)**

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

**References**

See Also

`as.methylumi`

Examples

```r
library(wateRmelon)
data(melon)
## pretend we have two different data sets
melon
pelon <- melon
sampleNames(pelon) <- gsub("6", 7, sampleNames(pelon))
combo(melon, pelon)
```

---

**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 exprmethy450 objects.

**Usage**

```r
dasen ( mns, uns, onetwo, fudge = 100, ret2=FALSE, ... )
nasen ( mns, uns, onetwo, ret2=FALSE, fudge = 100, ... )
betaqn( 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, ... )
daten1( mn, un, 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 MethyLumiSet, 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
danen 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 and unmethylated)
nasen same as naten but type I and type II intensities quantile normalized separately
tost method from Touleimat and Tost 2011
fucks 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.
dasen-methods

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

References

See Also
pfilter, as.methylumi

Examples

#MethyLumiSet method
data(melon)
melon.dasen <- dasen(melon)

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

Arguments

\[ mn, mns, data, bn \]
A MethyLumiSet object. Sample names names are used to get Sentrix row and column by default, see `...'.

\[ fudge \]
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 el 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 ) naten ( mn, fudge = 100 ) nanet ( mn, fudge = 100 ) nanes ( mn, fudge = 100 ) danes ( mn, fudge = 100, ... ) danet ( mn, fudge = 100, ... ) danen ( 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

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 typeI and type II background are equalised first.
- **danet**: same as nanet, except typeI and type II background are equalised first.
- **danen**: background equalisation only, no normalization.
- **daten1**: same as naten, except typeI 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 ) nanes ( mns, uns, onetwo, fudge = 100 ) danes ( mn, un, fudge = 100, ... ) danet ( mn, un, onetwo, fudge = 100, ... ) dann ( mns, uns, onetwo, fudge = 100, ... ) daten1( mn, un, onetwo, fudge = 100, ... ) daten2( mn, un, onetwo, fudge = 100, ... ) tost ( mn, un, da, pn ) fuks ( data, anno ) swan ( mn, un, qc )
```

**Author(s)**

Leonard.Schalkwyk@kcl.ac.uk

**References**


**db1**

*Internal watermelon functions for calculating betas*

**Description**

Db1 is used for quantile normalizing methylated together with unmethylated (dye bias methods nanet, nanes, danes and danet. dfs* functions are used for smoothing the background equalization in methods whose names start with d (daten etc).
**db1**

**Usage**

```r
db1(mn, un)
dfsfit(mn, onetwo, roco=unlist(data.frame(strsplit(colnames(mn), "_"),
stringsAsFactors = FALSE)[2, ]), ... )
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’
- `roco` 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).
- `...` no additional arguments currently used

**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.

**Value**

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

**Author(s)**

Leonard.Schalkwyk@kcl.ac.uk

**References**

**dmrse**

**Standard error of iDMR 450k array DNA methylation features**

**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.

**Usage**

```r
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 (methyumi 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**

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

#MethyLumiSet method after normalization
melon.dasen <- dasen(melon)
dmrse(melon.dasen)
```
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, Methy1Set, MethyLumiSe, or RGChannelSet) and calculate the metric.

---

**genki**

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 Methy1Set 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)
```

---

**genki-methods**

Methods for Function genki in Package watermelon

**Description**

Methods for function genki in package watermelon. Please see genki 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.

---

**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
geose - calculate between-sample SNP standard error
gcoms - calculate between-sample SNP mean-squared deviation
### got

**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**

```r
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


iDMR

Imprinting differentially methylated region probes of Illumina 450 arrays

Description

A character vector of 227 probes 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) except for swan.

**Usage**

```r
data(melon)
```

**Format**

MethyLumiSet with assayData containing 3363 features, 12 samples

**Source**


**Examples**

```r
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**

```r
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 sexistest, 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
outlyx

Value

dmrse_row  see dmrse_row


dmrse_col  see dmrse_col


dmrse  see dmrse


gcoms_a  see genki

gcose_a  see genki

gcoms_b  see genki

gcose_b  see genki

gcoms_c  see genki

gcose_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=FALSE, ...)

outlyx-methods

Arguments

x A MethyLumiSet, 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 principal 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.

Note

May perform poorly on normalized data

Author(s)

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

Examples

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

dput(outliers)

Methods

signature(x = "MethyLumiSet") all of the methods simply extract betas from the data object (which can be a MethylSet, 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 bead count. The pfilter function does not correct for this, but simply uses the detection p-value and bead count provided by each package.

Usage

pfilter(mn, un, bn, da, pn, bc, perCount=NULL, pnthresh = 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 #methylumi 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 bead count <3, or NULL when mn is a MethyLumiSet or RGChannelSetExtended object

**perCount**  
remove sites having this percentage of samples with a bead count <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

signature(mn = 'MethyLumiSet') This is used for performing the pfilter method on MethyLumiSet objects produced by methylumiR.

signature(mn = 'RGChannelSetExtended') This is used for performing the 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**  
*Probe-Wise Outlier Detection*

Description

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

Usage

pwod(object, mul=4)
## pwod

### Methods for Function `pwod` in Package `wateRmelon`

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

### signature

```r
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

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

A measure of Normalization Violence

### Description

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

| norm   | Matrix of normalized betas |
| raw    | Matrix of raw betas        |

Value

Returns data.frame containing rmse, 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

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.

Author(s)

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

References

methylumi

Examples

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

seabi

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)

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 pv, 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


sextest

Test Illumina methylation 450K array probes for sex difference (internal function for calculating seabi performance metric)

Description

This is a wrapper for lm 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 `lm`.

**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`

**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)
```

**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 =)
```
Arguments

mats
chans
parallel
protocol.data
IDAT
force
coef (agep), 4
colnames, MethyLumiSet-method
   (colnames-methods), 12
colnames-methods, 12
columnMatrix (wm_internal), 35
combo, 12
concatenateMatrices
   (adaptRefQuantiles), 3
cRankMatrices (adaptRefQuantiles), 3
correctI (Beta2M), 7
correctII (Beta2M), 7
danen (dasen), 13
danen, MethylSet-method
   (dasen-minfi-methods), 17
danen, MethyLumiSet-method
   (dasen-methods), 15
danen, RGChannelSet-method
   (dasen-minfi-methods), 17
danes (dasen), 13
danes, MethylSet-method
   (dasen-minfi-methods), 17
danes, MethyLumiSet-method
   (dasen-methods), 15
danes, RGChannelSet-method
   (dasen-minfi-methods), 17
daten (dasen), 13, 25
daten, MethylSet-method
   (dasen-minfi-methods), 17
daten, MethyLumiSet-method
   (dasen-methods), 15
daten, RGChannelSet-method
   (dasen-minfi-methods), 17
dansen-methods, 15
dansen-minfi-methods, 17
dataDetectPval2NA (adaptRefQuantiles), 3
DataToNCChannelSet2 (wm_internal), 35
daten1 (dasen), 13
daten1, MethylSet-method
   (dasen-minfi-methods), 17
daten1, MethyLumiSet-method
   (dasen-methods), 15
daten1, RGChannelSet-method
   (dasen-minfi-methods), 17
daten2 (dasen), 13
daten2, MethylSet-method
   (dasen-minfi-methods), 17
daten2, MethyLumiSet-method
   (dasen-methods), 15
daten2, RGChannelSet-method
   (dasen-minfi-methods), 17
db1, 18
designItoMandU2 (wm_internal), 35
designItoMandU2 (wm_internal), 35
detectionPvalfilter
   (adaptRefQuantiles), 3
dfort (wm_internal), 35
dfs2 (db1), 18
dfsfit (db1), 18
dmrse, 20, 21, 32
dmrse, exprmethy450-method
   (dmrse-methods), 21
dmrse, MethylSet-method (dmrse-methods), 21
dmrse, MethyLumiSet-method
   (dmrse-methods), 21
dmrse, RGChannelSet-method
   (dmrse-methods), 21
dmrse-methods, 21
dmrse_col, 25
dmrse_col (dmrse), 20
dmrse_col, exprmethy450-method
   (dmrse-methods), 21
dmrse_col, MethylSet-method
   (dmrse-methods), 21
dmrse_col, MethyLumiSet-method
   (dmrse-methods), 21
dmrse_col, RGChannelSet-method
   (dmrse-methods), 21
dmrse_col-methods (dmrse-methods), 21
dmrse_row (dmrse), 20
dmrse_row, exprmethy450-method
   (dmrse-methods), 21
dmrse_row, MethylSet-method
   (dmrse-methods), 21
dmrse_row, MethyLumiSet-method
   (dmrse-methods), 21
dmrse_row, RGChannelSet-method
   (dmrse-methods), 21
dmrse_row-methods (dmrse-methods), 21
epic.controls (readEPIC), 31
epic.ordering (readEPIC), 31
extractAssayDataFromList2
   (wm_internal), 35
filterXY (adaptRefQuantiles), 3
INDEX

findAnnotationProbes
(adaptRefQuantiles), 3
fot (got), 23
fus (dasen), 13
fus, exprmethyl450-method
(betaqn-exprmethyl450-methods), 8
fus, MethylSet-method
(dasen-minfi-methods), 17
fus, MethyLumiSet-method
(dasen-methods), 15
fus, RGChannelSet-method
(dasen-minfi-methods), 17
gcoms (genkme), 22
gcose, 32
gcose (genkme), 22
genall (wm_internal), 35
genki, 20, 21, 22, 23, 25
genki, exprmethyl450-method
(genki-methods), 22
genki, MethylSet-method (genki-methods), 22
genki, MethyLumiSet-method
(genki-methods), 22
genki, RGChannelSet-method
(genki-methods), 22
genki-methods, 22
genkme, 22
genkus (genkme), 22
genme (wm_internal), 35
genus (wm_internal), 35
getColumns (as.methylumi-methods), 5
getControlProbes2 (wm_internal), 35
getMethylationBeadMappers2
(wm_internal), 35
getMethyImIumBeta (adaptRefQuantiles), 3
getQuantiles (adaptRefQuantiles), 3
getSamples (adaptRefQuantiles), 3
getsnp (genkme), 22
goodSNP (wm_internal), 35
got, 23
IDATsToMatrices2 (wm_internal), 35
IDATtoMatrix2 (wm_internal), 35
iDMR, 24
iqrFun (wm_internal), 35
loadMethylumi2 (adaptRefQuantiles), 3
lumiMethyR2 (adaptRefQuantiles), 3
M2Beta (Beta2M), 7
melon, 25
mergeProbeDesigns2 (wm_internal), 35
methylumIDATepic (wm_internal), 35
methylumiR, 25
metrics, 25
mvFun (wm_internal), 35
nanes (dasen), 13
nanes, MethylSet-method
(dasen-minfi-methods), 17
nanes, MethyLumiSet-method
(dasen-methods), 15
nanes, RGChannelSet-method
(dasen-minfi-methods), 17
nanet (dasen), 13
nanet, MethylSet-method
(dasen-minfi-methods), 17
nanet, MethyLumiSet-method
(dasen-methods), 15
nanet, RGChannelSet-method
(dasen-minfi-methods), 17
naten (dasen), 13
naten, MethylSet-method
(dasen-minfi-methods), 17
naten, MethyLumiSet-method
(dasen-methods), 15
naten, RGChannelSet-method
(dasen-minfi-methods), 17
outlyx, 26
outlyx, MethylSet-method
(outlyx-methods), 27
outlyx, MethyLumiSet-method
(outlyx-methods), 27
outlyx, RGChannelSet-method
(outlyx-methods), 27
outlyx-methods, 27
oxyscale (wm_internal), 35
pcouted (wm_internal), 35
pfilter, 15, 28
pfilter, MethyLumiSet-method (pfilter), 28
pfilter, RGChannelSetExtended-method (pfilter), 28
pfilter-methods (pfilter), 28
pipelineIlluminaMethylation.batch (adaptRefQuantiles), 3
preprocessIlluminaMethylation (adaptRefQuantiles), 3
pwod, 29
pwod, MethylSet-method (pwod-methods), 30
pwod, MethyLumiSet-method (pwod-methods), 30
pwod, RGChannelSet-method (pwod-methods), 30
pwod-methods, 30
qual, 30
readEPIC, 31
referenceQuantiles (adaptRefQuantiles), 3
robustQuantileNorm_Illumina450K (adaptRefQuantiles), 3
seabi, 20, 25, 32, 33, 35
seabi, exprmethy450-method (seabi-methods), 33
seabi, MethylSet-method (seabi-methods), 33
seabi, MethyLumiSet-method (seabi-methods), 33
seabi, RGChannelSet-method (seabi-methods), 33
seabi-methods, 33
seabird, 34, 35
sextest, 34
summits (Beta2M), 7
swan, 5, 25
swan (dasen), 13
swan, MethylSet-method (dasen-minfi-methods), 17
swan, MethyLumiSet-method (dasen-methods), 15
swan, RGChannelSet-method (dasen-minfi-methods), 17
tost, 3
tost (dasen), 13
tost, MethylSet-method (dasen-minfi-methods), 17
tost, MethyLumiSet-method (dasen-methods), 15