Package ‘methylumi’

April 23, 2016

Type Package

Title Handle Illumina methylation data

Version 2.16.0

Date 2015-06-30

Author Sean Davis, Pan Du, Sven Bilke, Tim Triche, Jr., Moiz Bootwalla

Depends Biobase, methods, R (>= 2.13), scales, reshape2, ggplot2, matrixStats, FDb.InfiniumMethylation.hg19 (>= 2.2.0), minfi

Imports BiocGenerics, S4Vectors, IRanges, GenomeInfoDb, GenomicRanges, SummarizedExperiment, Biobase, graphics, lattice, annotate, genefilter, AnnotationDbi, minfi, stats4, illuminaio

Suggests lumi, lattice, limma, xtable, SQN, MASS, matrixStats, parallel, rtracklayer, Biostrings, methyAnalysis, TCGAMethylation450k, IlluminaHumanMethylation450kanno.ilmn12.hg19, FDb.InfiniumMethylation.hg18 (>= 2.2.0), Homo.sapiens, knitr

biocViews DNAMethylation, TwoChannel, Preprocessing, QualityControl, CpGIsland

Maintainer Sean Davis <sdavis2@mail.nih.gov>

Description This package provides classes for holding and manipulating Illumina methylation data. Based on eSet, it can contain MIAME information, sample information, feature information, and multiple matrices of data. An ``intelligent'' import function, methylumiR can read the Illumina text files and create a MethyLumiSet. methylumIDAT can directly read raw IDAT files from HumanMethylation27 and HumanMethylation450 microarrays. Normalization, background correction, and quality control features for GoldenGate, Infinium, and Infinium HD arrays are also included.

Collate AllGenerics.R MethyLumiSet-class.R MethyLumiM-class.R MoreGenerics.R Methods.R bgcorr.R coercions.R detectionpval.R featureFilter.R mclapply_replace.R methylData-class.R methylumIDAT.R methylumiCSV.R methylumiR.R normalization.R plotNegOob.R qc.probe.plot.R readIDAT2.R stripMethyLumiSet.R utilities.R varFilter.R

VignetteBuilder knitr

BugReports https://github.com/seandavi/methylumi/issues/new

License GPL-2

NeedsCompilation no

R topics documented:

methylumi-package *Handle Illumina methylation data*

Description

This package contains a class structure for handling methylation data from Illumina as well as utility functions for loading the data from files generated by Illumina. Normalization that attempts to correct for dye bias is also included.

Important data classes include: MethyLumiSet and MethyLumiQC, both of which are subsets of the MethyLumi class, which is a subset of the eSet class.

A worked example of the use of the package can be found by typing: openVignette().

A full listing of the available documentation can be obtained by typing help.start() and selecting methylumi from the Packages link or by typing library(help="methylumi").

If you use the methylumIDAT function or its out-of-band preprocessing mechanisms in your work, a citation to the paper "Low-level processing of Illumina Infinium DNA methylation beadarrays" by TJ Triche, DJ Weisenberger, D Van Den Berg, KD Siegmund, and PW Laird, Nucleic acids research, 2013, would be appreciated.

Details

Package:	methylumi
Type:	Package
License:	GPL

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

References

http://watson.nci.nih.gov/~sdavis/software/R

See Also

Biobase

CpGs *Data frame describing loci on the 27 and 450k arrays.*

Description

Data frame describing loci on the 27 and 450k arrays.

Usage

```
data(CpGs)
```

Examples

```
data(CpGs)
head(CpGs)
```

`estimateM` *Estimate methylation M-value matrix*

Description

Estimate methylation M-value matrix from MethyLumiM-class object or eSet-class object, which include methylated and unmethylated probe intensities

Usage

```
estimateM(methyLumiM, returnType=c("ExpressionSet", "matrix"), offset=100)
```

Arguments

`methyLumiM` MethyLumiM-class object or eSet-class object, which include methylated and unmethylated probe intensities

`returnType` determine whether return an ExpressionSet (MethyLumiM in this case) or matrix object

`offset` offset added to the methylated and unmethylated probe intensities when estimating the M-value

Details

M-value is the log2 ratio between Illumina methylated and unmethylated probe intensities. As variations of small intensities can cause big changes in the ratio estimation, so an offset is added to methylated and unmethylated probe intensities when estimating the M-value.

Please check the lumi package for more details of estimateM function.

Value

A MethyLumiM or matrix object of methylation M-value

Author(s)

Pan DU

References

Du, P., Zhang, X, Huang, C.C., Jafari, N., Kibbe, W.A., Hou, L., and Lin, S.M., (2010) ’Comparison of Beta-value and M-value methods for quantifying methylation levels by microarray analysis’, (under review)

extractBarcodeAndPosition *Extract the Barcode and Position Information from Sentrix ID*

Description

The sentrix IDs from an illumina sentrix array contain positional information that might be useful. This function simply extracts that information from the ID itself.

Usage

```
extractBarcodeAndPosition(sentrixids)
```

Arguments

sentrixids	A character vector of sentrix IDs that look like: 1632405013_R001_C001

Value

A data.frame with three columns:

sentrix	numeric, the sentrix ID
row	numeric, the sentrix row
column	numeric, the sentrix column

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

See Also

methylumiR

Examples

```
extractBarcodeAndPosition(c('12341234_R001_C001'))
```

featureFilter — *Annotation-based Filtering of Features (CpG sites) in a MethyLumiSet or MethyLumiM object*

Description

Features with insufficient annotation carry little value for the subsequent data analysis. The function `featureFilter` provides options of filtering features (CpG sites) from a `MethyLumiSet` (or `MethyLumiM`) object based on available annotation data.

Usage

```
featureFilter(eset, require.entrez=FALSE,
    require.GOBP=FALSE, require.GOCC=FALSE,
    require.GOMF=FALSE, exclude.ChrX=FALSE,
    require.closeToTSS=FALSE, range.DistToTSS=c(-500, 300),
    require.CpGisland=FALSE, ...)
```

Arguments

`eset` A `MethyLumiSet` or `MethyLumiM` object.

`require.entrez` If `TRUE`, filter out features without an Entrez Gene ID annotation.

`require.GOBP, require.GOCC, require.GOMF`
If `TRUE`, filter out features whose target genes are not annotated to at least one GO term in BP, CC and MF ontology, respectively.

`exclude.ChrX` If `TRUE`, filter out features in chromosome X to avoid gender effect.

`require.closeToTSS`
If `TRUE`, filter out features that are not close to transcription start site (TSS). Features without annotation of distance to TSS will also be removed. Can only used for GoldenGate platform.

`range.DistToTSS`
Ignored if `require.colseToTSS` is `FALSE`. A vector of numeric values of length 2, indicating the range of tolerable distance from transcription start site (TSS) in basepair (bp). If `require.clostToTSS` is `TRUE`, features whose distance to TSS falls outside this designated range will be removed. The default value is $c(-500, 300)$, where -500 represents the distance to TSS from the left and 300 the distance from the right.

`require.CpGisland`
If `TRUE`, filter out features that are not in CpG islands.

`...` Unused, but available for specializing methods.

Value

The function `featureFilter` returns a list consisting of:

`eset` The filtered `MethyLumiSet` or `MethyLumiM` object.

`filter.log` A list giving details of how many probe sets where removed for each annotation-based filtering step performed.

Author(s)

Chao-Jen Wong <cwon2@fhcrc.org>

References

R. Bourgon, R. Gentleman, W. Huber, *Independent filtering increases power for detecting differentially expressed genes*, PNAS, vol. 107, no. 21, pp:9546-9551.

See Also

nsFilter

---

getAssayDataNameSubstitutions

Return a data.frame of AssayData name substitutions.

Description

The Illumina methylation platforms use two distinct platforms, the "goldengate" platform and the "infinium" platform. Each of these uses different file formats as well as different assay techologies. To make the downstream data handling more straightforward and uniform between the two different systems, a simple mapping from the column names in the output files from the Illumina software is used to convert things from Red/Green or Cy5/Cy3 to unmethylated/methylated. This function simply returns that mapping.

Usage

```
getAssayDataNameSubstitutions()
```

Details

A file in the extdata directory called "substitutions.txt" contains two columns. The function loads this file and uses the first column as a match against column names in the data file (with the "sample part" removed). If matched, the second column gives the replacement.

Value

A data.frame with two columns, regex and replacement.

Author(s)

Sean Davis <seandavi@gmail.com>

Examples

```
getAssayDataNameSubstitutions()
```

methylData-class *Class* "methylData", *superclass for MethyLumiSet and MethyLumiM*

Description

A superclass (virtual) for MethyLumiSet and MethyLumiM.

Objects from the Class

A virtual Class: No objects may be created from it.

Methods

diagnostics `signature(x = "methylData")`: diagnostic plots of data

methylated.N `signature(object = "methylData")`: accessor for assayData element of the same name

methylated.N<- `signature(object = "methylData", value = "matrix")`: replace method for assayData element of the same name

plotNAs `signature(object = "methylData")`: ...

pval.detect `signature(object = "methylData")`: accessor for assayData element of the same name

pval.detect<- `signature(object = "methylData", value = "numeric")`: replace method for assayData element of the same name

unmethylated.N `signature(object = "methylData")`: accessor for assayData element of the same name

unmethylated.N<- `signature(object = "methylData", value = "matrix")`: replace method for assayData element of the same name

Author(s)

Tim Triche, Jr.

See Also

`MethyLumiSet`,`MethyLumiM`

Examples

```
showClass("methylData")
```

MethyLumi-accessors *methylumi accessors*

Description

These functions serve as getters and setters for information in methylumi classes.

Usage

```
betas(object)
pvals(object)
methylated(object)
unmethylated(object)
getHistory(object)
QCdata(object)
```

Arguments

`object`	an object of class MethyLumi or a subclass

Details

See the methods definitions in `MethyLumiSet` and `MethyLumiQC` for details.

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

See Also

`normalizeMethyLumiSet`, `MethyLumiSet`, `MethyLumiQC`, `eSet`

MethyLumi-class *The base class for storing Illumina Methylation data*

Description

This class inherits from `eSet` from the Biobase package and is used as a base class for the other two methylumi classes, `MethyLumiSet` and `MethyLumiQC`.

Objects from the Class

The `MethyLumi` class is a virtual class and is not meant to be instantiated. Instead, one should instantiate a `MethyLumiSet` or a `MethyLumiQC` object.

Slots

`assayData`**:** Object of class `"AssayData"`

`phenoData`**:** Object of class `"AnnotatedDataFrame"`

`featureData`**:** Object of class `"AnnotatedDataFrame"` that will hold the annotation columns from the Beadstudio output, if they are available.

`experimentData`**:** Object of class `"MIAME"`

`annotation`**:** Object of class `"character"`; note that this slot is not currently used, but may be used in the future to store the character name of the annotation package, if available.

`.__classVersion__`**:** Object of class `"Versions"`

Extends

Class `"eSet"`, directly. Class `"VersionedBiobase"`, by class "eSet", distance 2. Class `"Versioned"`, by class "eSet", distance 3.

Methods

pvals<- `signature(object = "MethyLumi", value = "matrix")`: Set the assayData slot of the same name and stores the P-values from BeadStudio

pvals `signature(object = "MethyLumi")`: Get the assayData slot of the same name

betas<- `signature(object = "MethyLumi", value = "matrix")`: Set the assayData slot of the same name and represents the methylation values for the samples, analogous to exprs() in gene expression data.

betas `signature(object = "MethyLumi")`: Get the assayData slot of the same name

methylated<- `signature(object = "MethyLumi", value = "matrix")`: Set the assayData slot that represents the Methylated single-channel signal

methylated `signature(object = "MethyLumi")`: Get the assayData slot that represents the Methylated single-channel signal

unmethylated<- `signature(object = "MethyLumi", value = "matrix")`: Set the assayData slot that represents the Unmethylated single-channel signal

unmethylated `signature(object = "MethyLumi")`: Get the assayData slot that represents the Unmethylated single-channel signal

controlTypes `signature(object = "MethyLumi"`: Find the unique control type beeds in the QCdata slot.

qcplot `signature(object = "MethyLumi",what,...)`: Plot of QC data. This plot can be useful for diagnosing the problems associated with specific samples or arrays. The value for "what" is one of the control types (which can be found by using `controlTypes()` on the object.

summary `signature(object = "MethyLumi",...)`: summary method for MethyLumi objects.

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

See Also

methylumiR, MethyLumiSet, MethyLumiQC, eSet

Examples

```
## The class structure
showClass("MethyLumi")
## read in some data
## Read in sample information
samps <- read.table(system.file("extdata/samples.txt",
                                package = "methylumi"),sep="\t",header=TRUE)
## Perform the actual data reading
## This is an example of reading data from a
## Sentrix Array format file (actually two files,
## one for data and one for QC probes)
mldat <- methylumiR(system.file('extdata/exampledata.samples.txt',
        package='methylumi'),
      qcfile=system.file('extdata/exampledata.controls.txt',
        package="methylumi"),
      sampleDescriptions=samps)
mldat
## Get history information
getHistory(mldat)
## Get QC data, which is another eSet-derived object
QCdata(mldat)
```

MethyLumi-strippers *Strip excessive probe-level data from MethyLumiSets*

Description

450k datasets with probe-level stderrs, out-of-band intensities, and bead numbers can become huge. These functions help to manage their growth in memory, at least until preprocessing and QC is completed, whereupon the summary data can be exported to a RangedData-based object of some sort for integration.

Usage

```
stripMethyLumiSet(object)
stripBeadNs(object)
stripBeadSDs(object)
stripOOB(object)
```

Arguments

`object`	an object of class MethyLumi or a subclass

Author(s)

Tim Triche, Jr. <tim.triche@gmail.com>

methylumIDAT *methylumIDAT*

Description

Read a directory of methylumi idat files and return a `MethylumiSet`.

Usage

```
methylumIDAT(barcodes = NULL, pdat = NULL, parallel = F, n = F, n.sd =
F, oob = T, idatPath=getwd(), ...)
```

Arguments

`barcodes`	A vector of barcodes to read. Either this argument or `pdat` must be specified.
`pdat`	A data.frame describing the samples. A special column named "barcodes" can be used to specify the barcodes to be read.
`parallel`	If TRUE, an attempt will be made to process using multiple cores on a multicore machine.
`n`	Keep the bead numbers? (Default: no)
`n.sd`	Keep the bead-level SD? (Default: no)
`oob`	Keep the out-of-band (OOB) or opposite-channel signals? (Default: yes)
`idatPath`	The path to the directory containing the idat files.
`...`	Additional arguments to be passed to sub-functions.

Details

Read a set of .idat files and return a `MethylumiSet` object. If you use this function to any significant degree in your analysis, we would appreciate your citing the paper describing it, "Low-level processing of Illumina Infinium DNA methylation beadarrays", TJ Triche, DJ Weisenberger, D Van Den Berg, KD Siegmund, and PW Laird, Nucleic acids research, 2013.

Value

A `MethylumiSet` object.

Author(s)

Tim Triche, Jr.

See Also

The “methylumi450k“ vignette: vignette("methylumi450k", package="methylumi")

Examples

```
## Not run:
if(require('IlluminaHumanMethylation450k.db')) {
  barcodes <- c('6005486014_R04C02',
                '6005486023_R05C01')
  lumi450k <- methylumIDAT(barcodes,idatPath=system.file('extdata/idat',package='methylumi')) # no normalizatio
  sampleNames(lumi450k) <- c('TCGA1','TCGA2')
  show(lumi450k)
}

## End(Not run)
```

`methylumiGenerics` *Generics defined in methylumi*

Description

See the individual classes for details of methods.

Author(s)

Sean Davis, Pan Du, and Tim Triche, Jr.

`MethyLumiM-class` *Class "MethyLumiM": for Illumina Methylation microarray data using logRatios*

Description

MethyLumiM is a class inherited from `ExpressionSet-class`. It is designed for Illumina Methylation microarray data. The exprs dataMatrix included in the assayData slot of MethyLumiM object includes a matrix of M-values, which is the log2 ratio of methylated and unmethylated probe intensities. The MethyLumiM class include a boxplot function uniquely designed for two-mode histogram data. It also include a coerce function to map from `MethyLumi-class`, `MethyLumiSet-class` or other `eSet-class` inherited object to MethyLumiM class object.

Objects from the Class

Objects can be created by calls of the form `new("MethyLumiM", exprs, methylated, unmethylated, detection, methy` The "exprs" is a matrix of M-values, which is the log2 ratio of methylated and unmethylated probe intensities; "methylated" and "unmethylated" are intensity matrix measured by methylated and unmethylalted probes of Illumina Infinium methylation microarray; "detection" is the detection p-value outputted by Illumina GenomeStudio software; "methylated.N" and "unmethylated.N" are bead numbers for methylated and unmethylalted probes. "exprs", "methylated" and "unmethylated" information are required for MethyLumiM class. When creating a new MethyLumiM object, the information of "exprs", "methylated", "unmethylated" and "detection" can also be provided directly through "assayData".

Slots

`history`: Object of class `"data.frame"` recording the operation history of the LumiBatch object.

`controlData`: Object of class `"MethyLumiQC"` to keep the QC probe measurement information.

`dataType`: The type of data stored in the "exprs" data matrix in "assayData". It can be "M" (M-value), "Beta" (Beta-value") or "Intensity" (Intensity of CpG-site)

`assayData`: Object of class `"AssayData"`, which includes "exprs", "methylated", "unmethylated", "detection", "methylated.N" and "unmethylated.N" data matrix

`phenoData`: Object of class `"AnnotatedDataFrame"`, See `eSet-class`

`featureData`: Object of class `"AnnotatedDataFrame"`, See `eSet-class`

`experimentData`: Object of class `"MIAME"`, See `eSet-class`

`annotation`: Object of class `"character"`, See `eSet-class`

`protocolData`: Object of class `"AnnotatedDataFrame"`, See `eSet-class`

`.__classVersion__`: Object of class `"Versions"`, See `eSet-class`

Extends

Class `"ExpressionSet"`, directly. Class `"eSet"`, by class "ExpressionSet", distance 2. Class `"VersionedBiobase"`, by class "ExpressionSet", distance 3. Class `"Versioned"`, by class "ExpressionSet", distance 4.

Methods

boxplot `signature(x = "MethyLumiM")`: plot distribution of M-value

coerce `signature(from = "eSet", to = "MethyLumiM")`: map from `MethyLumi-class`, `MethyLumiSet-class` or other `eSet-class` inherited object to MethyLumiM class object. MethyLumiM object will only keep "exprs", "methylated", "unmethylated" and "detection" data matrix in the assayData.

getHistory `signature(object = "MethyLumiM")`: access the operation history of `MethyLumiM` object.

initialize `signature(.Object = "MethyLumiM")`: class initialization

methylated `signature(object = "MethyLumiM")`: retrieve the data matrix measured by methylated probes

methylated<- `signature(object = "MethyLumiM")`: set the data matrix measured by methylated probes

unmethylated `signature(object = "MethyLumiM")`: retrieve the data matrix measured by unmethylated probes

unmethylated<- `signature(object = "MethyLumiM")`: set the data matrix measured by unmethylated probes

methylated.N `signature(object = "MethyLumiM")`: retrieve the data matrix keeping the number of beads of methylated probes

methylated.N<- `signature(object = "MethyLumiM")`: set the data matrix keeping the number of beads of methylated probes

unmethylated.N signature(object = "MethyLumiM"): retrieve the data matrix keeping the number of beads of unmethylated probes

unmethylated.N<- signature(object = "MethyLumiM"): set the data matrix keeping the number of beads of unmethylated probes

detection signature(object = "MethyLumiM"): retrieve detection data matrix in AssayData-class

detection<- signature(object = "MethyLumiM"): set detection data matrix in AssayData-class

controlData signature(object = "MethyLumiM"): retrieve the controlData in MethyLumiQC-class

controlData<- signature(object = "MethyLumiM"): set controlData in MethyLumiQC-class

dataType signature(object = "MethyLumiM"): retrieve the dataType, by default it is "M", it can also be "Beta" or "Intensity"

dataType<- signature(object = "MethyLumiM"): set dataType in MethyLumiM-class, the value can be "M", "Beta" or "Intensity"

Author(s)

Pan DU

References

1. Du, P., Zhang, X, Huang, C.C., Jafari, N., Kibbe, W.A., Hou, L., and Lin, S.M., (2010) 'Comparison of Beta-value and M-value methods for quantifying methylation levels by microarray analysis'

See Also

MethyLumi-class and MethyLumiSet-class

Examples

```
showClass("MethyLumiM")
```

MethyLumiQC-class *Class* "MethyLumiQC" *for holding Illumina methylation QC data*

Description

This class inherits from the MethyLumi class (and therefore, from eSet in Biobase) and is designed to hold QC data from Illumina Beadstudio output. These data can be potentially useful when determining the cause for quality problems.

Objects from the Class

Objects can be created by calls of the form new("MethyLumiQC", assayData, phenoData, featureData, experimentDa

Slots

`assayData`**:** Object of class "`AssayData`"

`phenoData`**:** Object of class "`AnnotatedDataFrame`"

`featureData`**:** Object of class "`AnnotatedDataFrame`" containing the annotation columns from the Illumina Beadstudio output. In particular, the names of the probes describe the types of control probes.

`experimentData`**:** Object of class "`MIAME`"

`annotation`**:** Object of class "`character`", not currently used

`.__classVersion__`**:** Object of class "`Versions`"

Extends

Class "`MethyLumi`", directly. Class "`eSet`", by class "MethyLumi", distance 2. Class "`VersionedBiobase`", by class "MethyLumi", distance 3. Class "`Versioned`", by class "MethyLumi", distance 4.

Methods

initialize `signature(.Object = "MethyLumiQC")`

Cy3.N `signature(object = "MethyLumiQC")`: ...

Cy3<- `signature(object = "MethyLumiQC", value = "matrix")`: ...

Cy5.N `signature(object = "MethyLumiQC")`: ...

Cy5<- `signature(object = "MethyLumiQC", value = "matrix")`: ...

QCdata<- `signature(object = "MethyLumiSet", value = "MethyLumiQC")`: ...

combine `signature(x = "MethyLumiQC", y = "MethyLumiQC")`: ...

controlData<- `signature(object = "MethyLumiSet", value = "MethyLumiQC")`: ...

controlTypes `signature(object = "MethyLumiQC")`: determine the character vector of control types from the QCdata information

hist `signature(x = "MethyLumiQC")`: ...

intensitiesByChannel `signature(object = "MethyLumiQC")`: ...

methylated `signature(object = "MethyLumiQC")`: ...

negctls.stderr `signature(object = "MethyLumiQC", channel = "character")`: ...

negctls.stderr `signature(object = "MethyLumiQC", channel = "missing")`: ...

negctls `signature(object = "MethyLumiQC", channel = "character")`: ...

negctls `signature(object = "MethyLumiQC", channel = "missing")`: ...

negnorm `signature(object = "MethyLumiQC", channel = "character")`: ...

negnorm `signature(object = "MethyLumiQC", channel = "missing")`: ...

normctls `signature(object = "MethyLumiQC")`: ...

qcplot `signature(object = "MethyLumiQC", what, ...)`: QC plots of various controltypes

unmethylated `signature(object = "MethyLumiQC")`: ...

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

See Also

`methylumiR`, `MethyLumiSet`, `MethyLumi`, `eSet`

Examples

```
showClass("MethyLumiQC")
```

`methylumiR` *Load data from Illumina methylation platform*

Description

This function is useful for loading Illumina methylation data into a MethyLumiSet object. Sample information can be supplied and will then be incorporated into the resulting phenoData slot.

Usage

```
methylumiR(filename, qcfile=NULL, sampleDescriptions = NULL, sep = NULL, ...)
```

Arguments

`filename` A filename of the excel-like file from BeadStudio

`qcfile` A filename of the excel-like file from BeadStudio

`sampleDescriptions`
A data.frame that contains at least one column, SampleID (case insensitive). This column MUST match the part of the column headers before the `.Avg_Beta`, etc. Also, if a column called SampleLabel (case insensitive), it is used for sample labels, IF the sampleLabel column contains unique identifiers

`sep` seperator used in the BeadStudio (or GenomeStudio) output file. If it is NULL, the function will automatically estimate it.

`...` Passed into `read.delim()`

Details

Used to construct a MethyLumiSet object....

Value

A MethyLumiSet object

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

See Also

MethyLumiSet-class, MethyLumiQC-class

Examples

```
## Read in sample information
samps <- read.table(system.file("extdata/samples.txt",
                                package = "methylumi"),sep="\t",header=TRUE)
## Perform the actual data reading
## This is an example of reading data from an
## Sentrix Array format file (actually two files,
## one for data and one for QC probes)
mldat <- methylumiR(system.file('extdata/exampledata.samples.txt',package='methylumi'),
                qcfile=system.file('extdata/exampledata.controls.txt',package="methylumi"),
                    sampleDescriptions=samps)
mldat
```

MethyLumiSet-class *Class* "MethyLumiSet" *for containing Illumina methylation data*

Description

This class inherits from the MethyLumi class (and therefore, from eSet in Biobase) and is designed to hold both the intensities and the calculated betas, as well as pvalues if present.

Objects from the Class

Objects can be created by calls of the form `new("MethyLumiSet", assayData, phenoData, featureData, experimentD` An object of this type is the main storage class for methylation data from Illumina. Subsetting, etc., works as normal (rows represent genes, columns represent samples). There is also a rudimentary history tracking system that is modeled after that from the lumi package.

Slots

`QC`: Object of class `"QCDataOrNULL"`, containing either the `MethyLumiQC` object or NULL

`history`: Object of class `"data.frame"`, containing a running history of transforms to the data contained herein

`assayData`: Object of class `AssayData`

`phenoData`: Object of class `AnnotatedDataFrame`

`featureData`: Object of class `AnnotatedDataFrame`, containing the annotation columns from the Illumina Beadstudio output

`experimentData`: Object of class `MIAME`

`annotation`: Object of class `"character"`, not currently used

`.__classVersion__`: Object of class `"Versions"`

`protocolData`: Object of class `"AnnotatedDataFrame"` that contains protocol information, including scan date if available

Extends

Class "`MethyLumi`", directly. Class "`methylData`", directly. Class "`eSet`", by class "MethyLumi", distance 2. Class "`VersionedBiobase`", by class "MethyLumi", distance 3. Class "`Versioned`", by class "MethyLumi", distance 4.

Methods

[`signature(x = "MethyLumiSet")`: subsetting, genes as rows, samples as columns

betas<- `signature(object = "MethyLumiSet", value = "matrix")`: Set the assayData slot of the same name

betas `signature(object = "MethyLumiSet")`: Get the assayData slot of the same name

boxplot `signature(x = "MethyLumiSet")`: boxplot of all sample betas

combine `signature(x = "MethyLumiSet", y = "MethyLumiSet")`

corplot `signature(x = "MethyLumiSet")`

exprs `signature(object = "MethyLumiSet")`: returns m-values

getHistory `signature(object = "MethyLumiSet")`: returns a data.frame containing the history for this object

hist `signature(x = "MethyLumiSet")`: histogram of the betas for the data

initialize `signature(.Object = "MethyLumiSet")`

pairs `signature(x = "MethyLumiSet")`: pairs plot of the betas for the object. Note that pairs plots of more than a few samples are not very useful.

plotSampleIntensities `signature(x = "MethyLumiSet")`: The intensities as output by the Beadstudio software often show a considerable amount of dye bias. This method shows a graphical example of this dye bias. In short, for each of the Cy3 and Cy5 channels, a cutoff in beta is used to calculate which Cy3 and Cy5 values should be plotted at high-methylation and low-methylation status. Any offset between Cy3 and Cy5 when plotted in this way likely represents dye bias and will lead to biases in the estimate of beta.

QCdata<- `signature(object = "MethyLumiSet", value = "MethyLumiQC")`: assign QC data to the QC slot

QCdata `signature(object = "MethyLumiSet")`: retrieve the QC data.

show `signature(object = "MethyLumiSet")`

methylated<- `signature(object = "MethyLumiSet", value = "matrix")`: Set the assayData slot associated with methylated intensity

methylated `signature(object = "MethyLumiSet")`: Get the assayData slot associated with methylated intensity

unmethylated<- `signature(object = "MethyLumiSet", value = "matrix")`: Set the assayData slot associated with unmethylated intensity

unmethylated `signature(object = "MethyLumiSet")`: Get the assayData slot associated with unmethylated intensity

qcplot `signature(object = "MethyLumiSet", what, ...)`: QC plots of various controltypes

controlTypes `signature(object = "MethyLumiSet")`: determine the character vector of control types from the QCdata information

Cy3.N `signature(object = "MethyLumiSet")`: ...

Cy5.N `signature(object = "MethyLumiSet")`: ...

combine27k450k `signature(x = "MethyLumiSet", y = "MethyLumiSet")`: ...

controlData `signature(object = "MethyLumiSet")`: ...

controlData<- `signature(object = "MethyLumiSet", value = "MethyLumiQC")`: ...

featureFilter `signature(eset = "MethyLumiSet")`: ...

intensities.IB `signature(x = "MethyLumiSet", channel = "character")`: ...

intensities.IB `signature(x = "MethyLumiSet", channel = "missing")`: ...

intensities.M `signature(x = "MethyLumiSet", channel = "character")`: ...

intensities.M `signature(x = "MethyLumiSet", channel = "missing")`: ...

intensities.OOB.allelic `signature(x = "MethyLumiSet", channel = "character", allele = "character")`: ...

intensities.OOB.allelic `signature(x = "MethyLumiSet", channel = "missing", allele = "missing")`: ...

intensities.OOB `signature(x = "MethyLumiSet", channel = "character")`: ...

intensities.OOB `signature(x = "MethyLumiSet", channel = "missing")`: ...

intensities.U `signature(x = "MethyLumiSet", channel = "character")`: ...

intensities.U `signature(x = "MethyLumiSet", channel = "missing")`: ...

intensitiesByChannel `signature(object = "MethyLumiSet")`: ...

negctls.stderr `signature(object = "MethyLumiSet", channel = "character")`: ...

negctls.stderr `signature(object = "MethyLumiSet", channel = "missing")`: ...

negctls `signature(object = "MethyLumiSet", channel = "character")`: ...

negctls `signature(object = "MethyLumiSet", channel = "missing")`: ...

negnorm `signature(object = "MethyLumiSet", channel = "character")`: ...

negnorm `signature(object = "MethyLumiSet", channel = "missing")`: ...

normctls `signature(object = "MethyLumiSet")`: ...

plotSampleIntensities `signature(x = "MethyLumiSet")`: ...

probeNAs `signature(object = "MethyLumiSet")`: ...

sampleNAs `signature(object = "MethyLumiSet")`: ...

total.intensity `signature(object = "MethyLumiSet")`: ...

varFilter `signature(eset = "MethyLumiSet")`: ...

Author(s)

Sean Davis & Tim Triche, Jr.

See Also

`methylumiR`, `normalizeMethyLumiSet`, `methylumIDAT`, `MethyLumiQC`, `eSet`

Examples

```
showClass("MethyLumiSet")
```

mldat *Example SAM format Illumina methylation dataset*

Description

This is an example MethyLumiSet object.

Usage

```
data(mldat)
```

Examples

```
data(mldat)
```

normalizeMethyLumiSet *Normalize a MethyLumiSet, accounting for dye bias*

Description

The Illumina GoldenGate methylation platform uses two colors, one to represent the unmethylated state and the other to represent the methylated state. This function corrects that dye bias and recalculates the betas based on the corrected intensities.

For HumanMethylation27 data, the function does nothing.

For HumanMethylation450 data, the function delegates to normalizeViaControls() the task of scaling red and green intensities against a reference array (chip) which defaults to the first chip in a set. The code to do this is based on code from the 'minfi' package and uses the built-in normalization controls to scale the channels of the samples, so that a consistent degree of dye bias is maintained for Infinium II probes across an experiment or set of experiments.

Usage

```
normalizeMethyLumiSet(x, beta.cuts = c(0.2, 0.8), mapfun = c("atan", "ratio"))
```

Arguments

x	A MethyLumiSet object
beta.cuts	Two numeric values with the first less than the second and between 0 and 1, representing the beta cutoffs that will be used when determining the median intensities to which to correct. See details below.
mapfun	Either "atan" or "ratio". See details below.

Details

For HumanMethylation450 data, the function delegates to normalizeViaControls() the task of scaling red and green intensities against a reference array (chip) which defaults to the first chip in a set. The code to do this is based on code from the 'minfi' package and uses the built-in normalization controls to scale the channels of the samples, so that a consistent degree of dye bias is maintained for Infinium II probes across an experiment or set of experiments. The remainder of the documentation below is specific to GoldenGate data.

The Illumina GoldenGate methylation platform uses two colors, one to represent the unmethylated state and the other to represent the methylated state. This function corrects that dye bias and recalculates the betas based on the corrected intensities.

As a first step, the medians for each of Cy3 and Cy5 are calculated at high and low betas, representing the (nearly) fully methylated state and the (nearly) fully unmethylated states. Values of Cy3 and Cy5 that are negative are set to zero for this process. Then, the Cy5 medians are adjusted to match those of the Cy3 channel, thereby correcting the dye bias.

To map the new intensities back to betas, one of two map functions can be used. The default is the atan(Cy3/Cy5). The ratio maps using the function (Cy3/Cy3+Cy5). The differences should be very small, but we feel that the atan map function is probably the mathematically appropriate way of doing this.

Value

A new "MethyLumiSet" that contains the corrected betas and the adjusted intensities.

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

Examples

```
## Read in sample information
samps <- read.table(system.file("extdata/samples.txt",
                                package = "methylumi"),sep="\t",header=TRUE)
## Perform the actual data reading
## This is an example of reading data from an
## Sentrix Array format file (actually two files,
## one for data and one for QC probes)
mldat <- methylumiR(system.file('extdata/exampledata.samples.txt',package='methylumi'),
                qcfile=system.file('extdata/exampledata.controls.txt',package="methylumi"),
                    sampleDescriptions=samps)
mldatnorm <- normalizeMethyLumiSet(mldat)
```

plotSampleIntensities *Plot the sample intensities.*

Description

The Illumina methylation platforms all show a significant dye bias. The plotSampleIntensities method shows the density plots for the two channels allowing direct visualization of the effect.

Usage

```
plotSampleIntensities(x,beta.cuts,s)
```

Arguments

`x`	an object of class MethyLumi or a subclass
`beta.cuts`	cutoffs for low and high beta values
`s`	sample number to plot

Examples

```
data(mldat)
plotSampleIntensities(mldat,s=1)
```

`qcplot` *Methods for dealing with control data for Illumina methylation data.*

Description

The `qcplot` function simply generates a plot of the control probe information for a given `controlType`.

Usage

```
qcplot(object,controltype,...)
controlTypes(object,...)
```

Arguments

`object`	An object of class `MethyLumiSet` or `MethyLumiQC`
`controltype`	A single character value representing the bead type to plot from the quality control data. The available types are accessible via the `controlTypes` method.
`...`	passed to plot function

Details

The descriptions of the various control types can be obtained from the Illumina methylation user's guides.

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

See Also

`MethyLumiSet`, `MethyLumiQC`

Examples

```
data(mldat)
controlTypes(mldat)
qcplot(mldat,controlTypes(mldat)[3])
```

varFilter *Variation-based Filtering of Features (CpG sites) in a MethyLumiSet or MethyLumiM object*

Description

The function `varFilter` removes features exhibiting little variation across samples. Such non-specific filtering can be advantageous for downstream data analysis.

Usage

```
varFilter(eset, var.func=IQR, var.cutoff=0.5, filterByQuantile=TRUE, ...)
```

Arguments

`eset` An `MethyLumiSet` or `MethyLumiM` object.

`var.func` The function used as the per-feature filtering statistics.

`var.cutoff` A numeric value indicating the cutoff value for variation. If `filterByQuantile` is `TRUE`, features whose value of `var.func` is less than `var.cutoff`-quantile of all `var.func` value will be removed. It `FALSE`, features whose values are less than `var.cutoff` will be removed.

`filterByQuantile` A logical indicating whether `var.cutoff` is to be interpested as a quantile of all `var.func` (the default), or as an absolute value.

`...` Unused, but available for specializing methods.

Details

This function is a counterpart of functions `nsFilter` and `varFilter` available from the `genefilter` package. See R. Bourgon et. al. (2010) and `nsFilter` for detail.

It is proven that non-specific filtering, for which the criteria does not depend on sample class, can increase the number of discoverie. Inappropriate choice of test statistics, however, might have adverse effect. `limma`'s moderated t-statistics, for example, is based on empirical Bayes approach which models the conjugate prior of gene-level variance with an inverse of χ^2 distribution scaled by observed global variance. As the variance-based filtering removes the set of genes with low variance, the scaled inverse χ^2 no longer provides a good fit to the data passing the filter, causing the `limma` algorithm to produce a posterior degree-of-freedom of infinity (Bourgon 2010). This leads to two consequences: (i) gene-level variance estimate will be ignore, and (ii) the p-value will be overly optimistic (Bourgon 2010).

Value

The function `featureFilter` returns a list consisting of:

`eset` The filtered `MethyLumiSet` or `MethyLumiM` object.

`filter.log` Shows many low-variant features are removed.

Author(s)

Chao-Jen Wong <`cwon2@fhcrc.org`>

References

R. Bourgon, R. Gentleman, W. Huber, *Independent filtering increases power for detecting differentially expressed genes*, PNAS, vol. 107, no. 21, pp:9546-9551, 2010.

See Also

`nsFilter`

Examples

```
data(mldat)
## keep top 75 percent
filt <- varFilter(mldat, var.cutoff=0.25)
filt$filter.log
dim(filt$eset)
```

Index