oligo

April 19, 2009

Description

A class for Platform Design Information objects, stored using a database approach

Objects from the Class

Objects can be created by calls of the form `new("DBPDInfo", ...)`.

Slots

getdb: Object of class `"function"`

tableInfo: Object of class `"data.frame"`

manufacturer: Object of class `"character"`

genomebuild: Object of class `"character"`

geometry: Object of class `"integer"` with length 2 (rows x columns)

Extends

Class "PDInfo", directly.

Methods

No methods defined with class "DBPDInfo" in the signature.

Examples

```
##---- Should be DIRECTLY executable !! ----
```

`FeatureSet-class`   *Classes for Feature level data*

Description

Virtual class to store feature level data.

Objects from the Class

Objects can be created by calls of the form `new("FeatureSet", assayData, manufacturer, platform, exprs, phenoData, featureData, experimentData, annotation, ...)`.

Slots

manufacturer: Object of class `"character"`

platform: Object of class `"character"`

assayData: Object of class `"AssayData"`

phenoData: Object of class `"AnnotatedDataFrame"`

featureData: Object of class `"AnnotatedDataFrame"`

experimentData: Object of class `"MIAME"`

annotation: Object of class `"character"`

.__classVersion__: Object of class `"Versions"`

Extends

Class eSet, directly. Class VersionedBiobase, by class "eSet", distance 2. Class link[Biobase:class.Vers
by class "eSet", distance 3.

Methods

No methods defined with class "FeatureSet" in the signature.

See Also

eSet, VersionedBiobase, link[Biobase:class.Versioned]{Versioned}

Examples

```
##---- Should be DIRECTLY executable !! ----
```

```
ExonFeatureSet-class
```
"FeatureSet" Extensions

Description

Classes to store data from Expression/Exon/SNP/Tiling arrays at the feature level.

Objects from the Class

Objects can be created by calls of the form `new("ExonFeatureSet", assayData, manufacturer, platform, exprs, phenoData, featureData, experimentData, annotation, ...)`.

Slots

manufacturer: Object of class `"character"`

platform: Object of class `"character"`

assayData: Object of class `"AssayData"`

phenoData: Object of class `"AnnotatedDataFrame"`

featureData: Object of class `"AnnotatedDataFrame"`

experimentData: Object of class `"MIAME"`

annotation: Object of class `"character"`

.__classVersion__: Object of class `"Versions"`

Extends

Class `FeatureSet`, directly. Class `eSet`, by class "FeatureSet", distance 2. Class `VersionedBiobase`, by class "FeatureSet", distance 3. Class `link[Biobase:class.Versioned]{Versioned}`, by class "FeatureSet", distance 4.

Methods

No methods defined with class "ExonFeatureSet" in the signature.

See Also

`eSet`, `VersionedBiobase`, `link[Biobase:class.Versioned]{Versioned}`

Examples

```
##---- Should be DIRECTLY executable !! ----
```

---

mmindex   *Extract PM or MM indexes*

Description

Extracts the indexes for PM or MM probes, which can be used with the intensity matrix for subsetting.

Usage

```
mmindex(object, ...)
pmindex(object, ...)
```

Arguments

object	`AffySNPPDInfo`, `FeatureSet` or `platformDesign` object
...	extra arguments

---

mm   *Accessor to the PM/MM matrices*

Description

Accessor to the PM/MM matrices.

Usage

```
mm(object, subset = NULL)
pm(object, subset = NULL, ...)
mm(object)<-value
pm(object)<-value
```

Arguments

object	`FeatureSet` object
subset	not implemented yet
value	`matrix` or `BufferedMatrix` object
...	extra arguments

Details

The intensity matrix is stored as a `BufferedMatrix` object. This should leave more memory for computations.

Value

PM or MM matrix as requested as a `BufferedMatrix` object.

`MAplot-methods`   *MA plots*

Description

Create MA plots using a reference array (based on medians).

Methods

object = "FeatureSet" ExpressionFeatureSet

`PDInfo-class`   *Classes "PDInfo" and "SNPPDInfo"*

Description

Containers for chip information

Slots

manufacturer: Object of class `"character"`

genomebuild: Object of class `"character"`

geometry: Object of class `"integer"` with length 2 (rows x columns)

Methods

No methods defined with class "PDInfo" in the signature.

Examples

```
##---- Should be DIRECTLY executable !! ----
```

`listFeatureFields`   *List fields in the (pm/mm)feature table*

Description

Lists the fields in the (pm/mm)feature or featureSet table of the PDInfo object.

Usage

```
listFeatureFields(object)
listFeatureSetFields(object)
```

Arguments

`object`   `PDInfo` object

---

mmSequence   *Probe Sequeces*

Description

Accessor to the probe sequences.

Usage

```
mmSequence(object)
pmSequence(object, ...)
```

Arguments

object   `FeatureSet`, `platformDesign` or `AffySNPPDInfo` object

...   additional arguments

Value

A character vector with the PM or MM probe sequences.

SnpQSet-class   *Class "SnpQSet"*

Description

Class to store Snp (Quantifications) Summaries

Objects from the Class

Objects can be created by calls of the form `new("SnpQSet", assayData, senseThetaA, senseThetaB, antisenseThetaA, antisenseThetaB, phenoData, featureData, experimentData, annotation)`.

Slots

assayData: Object of class `"AssayData"`

phenoData: Object of class `"AnnotatedDataFrame"`

featureData: Object of class `"AnnotatedDataFrame"`

experimentData: Object of class `"MIAME"`

annotation: Object of class `"character"`

.__classVersion__: Object of class `"Versions"`

Extends

Class eSet, directly. Class VersionedBiobase, by class "eSet", distance 2. Class `link[Biobase:class.Vers`
by class "eSet", distance 3.

Methods

No methods defined with class "SnpQSet" in the signature.

See Also

[eSet](), [VersionedBiobase](), link[Biobase:class.Versioned]{Versioned}

Examples

```
##---- Should be DIRECTLY executable !! ----
```

---

getA   *Compute average log-intensities / log-ratios*

Description

Methods to compute average log-intensities and log-ratios across alleles, within strand.

Usage

```
getA(object)
getM(object)
```

Arguments

object   SnpQSet object.

Details

SNPRMA summarizes the SNP information into 4 quantities (log2-scale):

antisenseThetaA antisense allele A

antisenseThetaB antisense allele B

senseThetaA sense allele A

senseThataB sense allele B

The average log-intensities are given by: `(antisenseThetaA+antisenseThetaB)/2` and `(senseThetaA+senseThetaB)/2`.

The average log-ratios are given by: `antisenseThetaA-antisenseThetaB` and `senseThetaA-senseThetaB`.

Value

A 3-dimensional array (SNP's x Samples x Strand) with the requested measure.

If a Genomewide array (like Affy SNP 6.0) is used, the strand dimension described above is dropped.

See Also

[snprma]()

SnpCallSet-class   *"SnpSet" Extensions*

Description

Class to store genotype calls / copy number estimates (and confidences) obtained via SNP arrays.

Objects from the Class

Objects can be created by calls of the form `new("SnpCallSet", assayData, featureData, phenoData, experimentData, annotation, calls, callsConfidence, ...)`.

Slots

assayData: Object of class `"AssayData"`

phenoData: Object of class `"AnnotatedDataFrame"`

featureData: Object of class `"AnnotatedDataFrame"`

experimentData: Object of class `"MIAME"`

annotation: Object of class `"character"`

.__classVersion__: Object of class `"Versions"`

Extends

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

Methods

No methods defined with class "SnpCallSet" in the signature.

See Also

`eSet`, `VersionedBiobase`, `link[Biobase:class.Versioned]{Versioned}`

Examples

```
##---- Should be DIRECTLY executable !! ----
```

---

basecontent   *Sequence Base Contents*

Description

Function to compute the amounts of each nucleotide in a sequence.

Usage

```
basecontent(seq)
```

Arguments

seq   character vector of length n containing a valid sequence (A/T/C/G)

Value

`matrix` with n rows and 4 columns with the counts for each base.

Examples

```
sequences <- c("ATATATCCCCG", "TTTCCGAGC")
basecontent(sequences)
```

---

boxplot   *Boxplot*

Description

Boxplot for observed (log-)intensities in a FeatureSet-like object (ExpressionFeatureSet, ExonFeatureSet, SnpFeatureSet, TilingFeatureSet).

Usage

```
boxplot(x, ...)
```

Arguments

x   a `FeatureSet`-like object

...   arguments to be passed to plot

See Also

hist, image

---

chromosome   *Accessor for chromosome information*

Description

Returns chromosome information.

Usage

```
pmChr(object)
```

Arguments

object   `TilingFeatureSet` or `SnpCallSet` object

Details

`chromosome()` returns the chromosomal information for all probes and `pmChr()` subsets the output to the PM probes only (if a TilingFeatureSet object).

Value

Vector with chromosome information.

---

crlmm   *Genotype Calls*

Description

Performs genotype calls via CRLMM (Corrected Robust Linear Model with Maximum-likelihood based distances).

Usage

```
crlmm(filenames, outdir, batch_size=40000, balance=1.5,
     minLLRforCalls=c(5, 1, 5), recalibrate=TRUE,
     verbose=TRUE, pkgname, reference=TRUE)
justCRLMM(filenames, batch_size = 40000, minLLRforCalls = c(5, 1, 5),
recalibrate = TRUE, balance = 1.5, phenoData = NULL, verbose = TRUE,
pkgname = NULL, tmpdir=tempdir())
```

Arguments

filenames   character vector with the filenames.

outdir   directory where the output (and some tmp files) files will be saved.

batch_size   integer defining how many SNPs should be processed at a time.

recalibrate   Logical - should recalibration be performed?

balance   Control parameter to balance homozygotes and heterozygotes calls.

`minLLRforCalls`	
	Minimum thresholds for genotype calls.
`verbose`	Logical.
`phenoData`	`phenoData` object or `NULL`
`pkgname`	alt. pdInfo package to be used
`reference`	logical, defaulting to TRUE ...
`tmpdir`	Directory where temporary files are going to be stored at.

Value

`SnpCallSetPlus` object.

Examples

```
## crlmmResults <- justCRLMM(list.celfiles())
```

`featureIndex`	*Indexes for features in the expression matrix*

Description

Indexes for features (PM/MM/both) in the expression matrix.

Usage

```
featureIndex(object, which = c("both", "pm", "mm"), subset = NULL)
```

Arguments

`object`	`FeatureSet` object
`which`	subset of interest: "both", "pm" or "mm"?
`subset`	not implemented yet.

`featureNames-methods`
Retrieve featureNames

Description

Retrieve featureNames

---

genomeBuild *Genome Build Information*

Description

Returns the genome build information

Usage

```
genomeBuild(object)
```

Arguments

object   TilingFeatureSet or PDInfo object.

---

getCrlmmSummaries *Function to get CRLMM summaries saved to disk*

Description

This will read the summaries written to disk and return them to the user as a SnpCallSetPlus or SnpCnvCallSetPlus object.

Usage

```
getCrlmmSummaries(tmpdir)
```

Arguments

tmpdir   directory where CRLMM saved the results to.

Value

If the data were from SNP 5.0 or 6.0 arrays, the function will return a SnpCnvCallSetPlus object. It will return a SnpCallSetPlus object, otherwise.

---

getPlatformDesign   *Retrieve Platform Design object*

Description

Retrieve platform design object.

Usage

```
getPlatformDesign(object)
getPD(object)
```

Arguments

object   `FeatureSet` object

Value

`platformDesign` or `PDInfo` object.

---

hist   *Density estimate*

Description

Plot the density estimates for each sample

Usage

```
hist(x, ...)
```

Arguments

x   `FeatureSet` object

...   arguments to be passed to `lines`

---

image   *Display a Color Image*

Description

Produces an image (`graphics::image`) for each sample.

Usage

```
image(x, ...)
```

Arguments

x   `FeatureSet` object

...   parameters to be passed to `plot`

---

getChrXIndex   *Internals*

Description

Set of internal functions...

---

justSNPRMA   *Summarization of SNP data*

Description

This function implements the SNPRMA method for summarization of SNP data. It works directly with the CEL files, saving memory.

Usage

```
justSNPRMA(filenames, verbose = TRUE, phenoData = NULL, normalizeToHapmap = TRUE
```

Arguments

filenames   character vector with the filenames.

verbose   logical flag for verbosity.

phenoData   a `phenoData` object or `NULL`

normalizeToHapmap

     Normalize to Hapmap? Should always be TRUE, but it's kept here for future use.

Value

`SnpQSet` or a `SnpCnvQSet`, depending on the array type.

See Also

~~objects to See Also as [help](), ~~~

Examples

```
## snprmaResults <- justSNPRMA(list.celfiles())
```

`kind`	*Array type*

Description

Retrieves the array type.

Usage

```
kind(object)
```

Arguments

`object`   `FeatureSet` object

Value

String: "Expression", "Exon", "SNP" or "Tiling"

`list.celfiles`	*List CEL/XYS files*

Description

Lists the CEL (CEL.gz) / XYS files.

Usage

```
list.celfiles(...)
```

Arguments

`...`   parameters to be passed to `list.files`

Details

Character vector with the filenames.

---

manufacturer   *Accessor to the manufacturer slot.*

Description

Accessor to the manufacturer slot.

Usage

```
manufacturer(object)
manufacturer(object)<-value
```

Arguments

object   PDInfo or FeatureSet object

value   string, eg. "affymetrix" or "nimblegen"

---

nProbes   *Count number of probes*

Description

Counts the number of probes in a platformDesign object.

Usage

```
nProbes(object)
```

Arguments

object   platformDesign object

---

ncol   *~~function to do ... ~~*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
ncol(x)
nrow(x)
```

Arguments

x   platformDesign object

Details

Number of columns or rows of the array.

---

oligo-package   *The oligo package: a tool for analysis of oligonucleotide arrays*

Description

The **oligo** package handles oligonucleotide arrays: expression, tiling, SNP and exon chips. The supported manufacturers are Affymetrix and NimbleGen. The package provides tools for preprocessing.

Details

The package will read the raw intensity files (CEL for Affymetrix; XYS for NimbleGen) and allow the user to perform analyses starting at the feature-level.

Reading in the intensity files require the existence of data packages that contain the chip specific information (X/Y coordinates; feature types; sequence). These data packages packages are build using the **makePlatformDesign** package, except for the Affymetrix SNP arrays that uses the data packages built via **pdInfoBuilder** package. Soon, all the data packages will be built using the **pdInfoBuilder** package.

Author(s)

Benilton Carvalho - ⟨bcarvalh@jhsph.edu⟩

References

Carvalho, B.; Bengtsson, H.; Speed, T. P. & Irizarry, R. A. Exploration, Normalization, and Genotype Calls of High Density Oligonucleotide SNP Array Data. Biostatistics, 2006.

Examples

```
## rawData <- read.celfiles(list.celfiles())
## preProc <- snprma(rawData)
## gtCalls <- crlmm(preProc, correctionFile="outEM.rda")
```

---

platform   *Retrieve/replace platform information*

Description

Retrieve or replace the platform information.

Usage

```
platform(object)
platform(object)<-value
```

Arguments

object   FeatureSet object.

value   string

```
platformDesign-class
```
Class "platformDesign"

Description

Container for chip information, this is environment-based and is being replaced by DBPDInfo.

Objects from the Class

Objects can be created by calls of the form `new("platformDesign", ...)`.

Slots

featureInfo: Object of class `"environment"` containing vectors describing the array.

featureTypeDescription: Object of class `"list"`

type: Object of class `"character"` describing the type of the array ("expression", "exon", "SNP", "tiling"

nrow: Object of class `"numeric"` providing the number of rows.

ncol: Object of class `"numeric"` providing the number of columns

nwells: Object of class `"numeric"` providing the number of wells - NimbleGen specific.

lookup: Object of class `"data.frame"`: a lookup table for complex designs by NimbleGen.

indexes: Object of class `"list"` ~~

platforms: Object of class `"character"` ~~

manufacturer: Object of class `"character"` ~~

genomebuild: Object of class `"character"` ~~

Extends

Class `"PDInfo"`, directly.

Methods

No methods defined with class "platformDesign" in the signature.

Examples

```
##---- Should be DIRECTLY executable !! ----
```

---

plotM-methods *Methods for Log-Ratio plotting*

Description

The `plotM` methods are meant to plot log-ratios for different classes of data.

Methods

object = "SnpQSet", i = "character" Plot log-ratio for SNP data for sample i.

object = "SnpQSet", i = "integer" Plot log-ratio for SNP data for sample i.

object = "SnpQSet", i = "numeric" Plot log-ratio for SNP data for sample i.

object = "TilingQSet", i = "missing" Plot log-ratio for Tiling data for sample i.

---

pmAllele *Access the allele information for PM probes.*

Description

Accessor to the allelic information for PM probes.

Usage

```
pmAllele(object)
```

Arguments

object   `SnpFeatureSet` or `PDInfo` object.

---

pmPosition *Accessor to position information*

Description

`pmPosition` will return the genomic position for the (PM) probes.

Usage

```
pmPosition(object)
pmOffset(object)
```

Arguments

object   `AffySNPPDInfo`, `TilingFeatureSet` or `SnpCallSet` object

Details

`position` will return genomic position for all probes on a `FeatureSet` or `SnpCallSet` object.

`pmPosition` will return genomic position for PM probes on a tiling array.

`pmOffset` will return the offset information for PM probes on SNP arrays.

`pmStrand`   *Accessor to the strand information*

Description

Returns the strand information on SNP arrays for PM probes (0 - sense / 1 - antisense).

Usage

```
pmStrand(object)
```

Arguments

object   `AffySNPPDInfo` object

`probeNames`   *Accessor to feature names*

Description

Accessor to PM feature names.

Usage

```
probeNames(object, subset = NULL)
```

Arguments

object   `FeatureSet`, `AffySNPPDInfo` or `platformDesign` object.

subset   not implemented yet.

`read.celfiles` *Parser to CEL files*

Description

Reads CEL files.

Usage

```
read.celfiles(filenames, pkgname, phenoData, featureData, experimentData, notes
```

Arguments

`filenames`	a `character` vector with the CEL filenames.
`pkgname`	alternative data package to be loaded.
`phenoData`	phenoData
`featureData`	featureData
`experimentData`	
	experimentData
`notes`	notes
`verbose`	logical

Value

`this-is-escaped-codenormal-bracket28bracket-normal`
     if Expresssion arrays

`this-is-escaped-codenormal-bracket31bracket-normal`
     if Exon arrays

`this-is-escaped-codenormal-bracket34bracket-normal`
     if SNP arrays

`this-is-escaped-codenormal-bracket37bracket-normal`
     if Tiling arrays

See Also

[read.xysfiles](#) for NimbleGen arrays

---

read.xysfiles   *Parser to XYS files*

Description

NimbleGen provides XYS files which are read by this function.

Usage

```
read.xysfiles(filenames, pkgname = NULL, phenoData = NULL, featureData = NULL, e
```

Arguments

filenames	`character` vector with filenames.
pkgname	`character` vector with alternative PD Info package name
phenoData	phenoData
featureData	featureData
experimentData	
	experimentData
notes	notes
verbose	verbose

Details

The function will read the XYS files provided by NimbleGen Systems and return an object of class
FeatureSet.

Value

```
this-is-escaped-codenormal-bracket29bracket-normal
```
                if Expresssion arrays
```
this-is-escaped-codenormal-bracket32bracket-normal
```
                if Tiling arrays

See Also

[read.celfiles](#) for Affymetrix arrays

---

readSummaries   *Read summaries generated by crlmm*

Description

This function read the different summaries generated by crlmm.

Usage

```
readSummaries(type, tmpdir)
```

Arguments

type	type of summary of `character` class: 'alleleA', 'alleleB', 'alleleA-sense', 'alleleA-antisense', 'alleleB-sense', 'alleleB-antisense', 'calls', 'llr', 'conf'.
tmpdir	directory containing the output saved by crlmm

Details

On the 50K and 250K arrays, given a SNP, there are probes on both strands (sense and antisense). For this reason, the options 'alleleA-sense', 'alleleA-antisense', 'alleleB-sense' and 'alleleB-antisense' should be used **only** with such arrays (XBA, HIND, NSP or STY).

On the SNP 5.0 and SNP 6.0 platforms, this distinction does not exist in terms of algorithm (note that the actual strand could be queried from the annotation package). For these arrays, options 'alleleA', 'alleleB' are the ones to be used.

The options `calls`, `llr` and `conf` will return, respectivelly, the CRLMM calls, log-likelihood ratios (for devel purpose **only**) and CRLMM confidence calls matrices.

Value

Matrix with values of summaries.

---

rma   *RMA*

Description

RMA

Usage

```
rma(object, ...)
```

Arguments

object	FeatureSet object
...	the dots

Details

NOT recommended for SNP arrays, see `snprma`.

Value

`ExpressionSet` object.

See Also

`snprma`

`sequenceDesignMatrix`
                    Create design matrix for sequences

Description

Creates design matrix for sequences.

Usage

```
sequenceDesignMatrix(seqs)
```

Arguments

`seqs`   `character` vector of 25-mers.

Value

Matrix with length(seqs) rows and 75 columns.

`snprma`   *Preprocessing SNP Arrays*

Description

This function preprocess SNP arrays.

Usage

```
snprma(object, verbose = TRUE, normalizeToHapmap = TRUE)
```

Arguments

`object`   `SnpFeatureSet` object
`verbose`   Verbosity flag. `logical`
`normalizeToHapmap`
                 internal

Value

A `SnpQSet` object.

antisenseThetaA   *Summarized Measures for SNP*

Description

SNPRMA summarizes the intensities for each SNP into 4 quantities. These measures are at the log2 scale and are used to create log-ratios and log-averages, which are used by the genotype algorithm and copy number tool.

Usage

```
antisenseThetaA(object)
antisenseThetaB(object)
senseThetaA(object)
senseThetaB(object)
```

Arguments

object   SnpQSet object

Details

For a given SNP, an Affymetrix SNP array usually contains multiple features for each combination allele x strand, on which the summaries are computed.

Value

A matrix with R rows (SNP's) x C columns (Samples).

See Also

snprma, crlmm

Index