Package ‘oligoClasses’

March 26, 2015

Version 1.29.6

Title Classes for high-throughput arrays supported by oligo and crlmm

Author Benilton Carvalho and Robert Scharpf

Maintainer Benilton Carvalho <beniltoncarvalho@gmail.com> and Robert Scharpf <rscharpf@jhsph.edu>

Depends R (>= 2.14)

Imports BiocGenerics (>= 0.3.2), Biobase (>= 2.17.8), methods, graphics, IRanges (>= 2.1.10), GenomicRanges (>= 1.19.6), Biostrings (>= 2.23.6), affyio (>= 1.23.2), ff, foreach, BiocInstaller, utils, S4Vectors

Enhances doMC, doMPI, doSNOW, doParallel, doRedis

Suggests RSQLite, hapmapspn5, hapmapspn6, pd genomewidesnp6, pd genomewidesnp5, pd mapping50k hind240, pd mapping50k xba240, pd mapping250k sty, pd mapping250k nsp, genomewidesnp6Crlmm (>= 1.0.7), genomewidesnp5Crlmm (>= 1.0.6), RUnit, human370v1cCrlmm

Description This package contains class definitions, validity checks, and initialization methods for classes used by the oligo and crlmm packages.

License GPL (>= 2)

LazyLoad yes

utilis-parallel.R methods-gSet.R initialize-methods.R
methods-AlleleSet.R methods-AnnotationedDataFrame.R
methods-FeatureSet.R methods-AssayData.R
methods-SnpFeatureSet.R methods-oligoSnpSet.R
methods-CopyNumberSet.R methods-CNSet.R methods-PDInfo.R
methods-RangedDataCnv.R methods-SnpSet.R
methods-GenomeAnnotatedDataFrame.R methods-BeadStudioSet.R
methods-BeadStudioSetList.R methods-gSetList.R
methods-GRanges.R methods-SummarizedExperiment.R show-methods.R
functions.R zzz.R

biocViews Infrastructure
## Local Variables

## time-stamp-pattern
```
8/Date: %3a %3b %2d %02H:%02M:%02S %Z %:%y\n```

## End

NeedsCompilation no

### R topics documented:

- affyPlatforms
- AlleleSet-class
- annotationPackages
- AssayData-methods
- AssayDataList
- assayDataList-methods
- batch
- batchStatistics
- BeadStudioSet-class
- BeadStudioSetList-class
- celfileDate
- celfileName
- checkExists
- checkOrder
- chromosome-methods
- chromosome2integer
- CNSet-class
- CopyNumberSet-class
- CopyNumberSet-methods
- createFF
- db
- DBPDInfo-class
- efsExample
- exprs-methods
- featureDataList-methods
- FeatureSet-class
- ffdf-class
- ff_matrix-class
- ff_or_matrix-class
- fileConnections
- flags
- generics
- GenomeAnnotatedDataFrame-class
- GenomeAnnotatedDataFrameFrom-methods
- genomeBuild
- geometry
- getA
- getBar
- getSequenceLengths
R topics documented:

- GRanges-methods ................................................................. 36
- gSet-class ......................................................................... 38
- gSetList-class ..................................................................... 39
- i2p ................................................................................... 40
- initializeBigMatrix ............................................................ 41
- integerMatrix ...................................................................... 42
- is.ffmatrix .......................................................................... 43
- isPackageLoaded ............................................................... 44
- isSnp-methods ................................................................. 45
- kind .................................................................................. 45
- ldSetOptions ....................................................................... 46
- length-methods ............................................................... 47
- library2 ............................................................................. 47
- list.celfiles ....................................................................... 48
- ListClasses .......................................................................... 49
- locusLevelData .................................................................. 49
- makeFeatureGRanges .......................................................... 50
- manufacturer-methods ......................................................... 51
- ocLapply ........................................................................... 51
- ocSamples .......................................................................... 52
- oligoSet ............................................................................ 53
- oligoSnpSet-methods ............................................................ 54
- parStatus .......................................................................... 54
- pdPkgFromBioC .................................................................. 55
- platform-methods ............................................................ 56
- pmFragmentLength-methods .................................................. 56
- position-methods ............................................................. 56
- requireAnnotation ............................................................ 57
- requireClusterPkgSet .......................................................... 58
- sampleNames-methods ........................................................ 58
- scqsExample ................................................................. 59
- setCluster ........................................................................ 59
- sfsExample ................................................................. 60
- SnpSet-methods .............................................................. 61
- SnpSet2-class ................................................................. 62
- SnpSuperSet-class ............................................................. 63
- splitIndicesByLength ........................................................... 64
- sqsExample ................................................................. 65
- SummarizedExperiment-methods ........................................... 66

Index 67
AlleleSet-class

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provides a listing of available Affymetrix platforms currently supported by the R package oligo</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>affyPlatforms()</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A vector of class character.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Author(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>R. Scharpf</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>affyPlatforms()</td>
</tr>
</tbody>
</table>

AlleleSet-class

Class "AlleleSet"

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A class for storing the locus-level summaries of the normalized intensities</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects from the Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects can be created by calls of the form new(&quot;AlleleSet&quot;, assayData, phenoData, featureData, experimentData, annotation, protocolData, N__classVersion__)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Slots</th>
</tr>
</thead>
</table>
| 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" ~
| protocolData: Object of class "AnnotatedDataFrame" ~
| __classVersion__: Object of class "Versions" ~ |
annotationPackages

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

Methods

allele signature(object = "AlleleSet"): extract allele specific summaries. For 50K (XBA and Hind) and 250K (Sty and Nsp) arrays, an additional argument (strand) must be used (allowed values: 'sense', 'antisense').

bothStrands signature(object = "AlleleSet"): tests if data contains allele summaries on both strands for a given SNP.

bothStrands signature(object = "SnpFeatureSet"): tests if data contains allele summaries on both strands for a given SnpFeatureSet.

db signature(object = "AlleleSet"): link to database connection.

geta signature(object = "AlleleSet"): average intensities (across alleles)

getM signature(object = "AlleleSet"): log-ratio (Allele A vs. Allele B)

Author(s)
R. Scharpf

See Also

SnpSuperSet, CNSet

Examples

showClass("AlleleSet")
## an empty AlleleSet
x <- new("matrix")
new("AlleleSet", senseAlleleA=x, senseAlleleB=x, antisenseAlleleA=x, antisenseAlleleB=x)
## or
new("AlleleSet", alleleA=x, alleleB=x)

annotationPackages

Annotation Packages

Description

annotationPackages will return a character vector of the names of annotation packages.

Usage

annotationPackages()

Value

a character vector of the names of annotation packages
Description

Batch statistics used for estimating copy number are stored as AssayData in the 'batchStatistics' slot of the CNSet class. Each element in the AssayData must have the same number of rows and columns. Rows correspond to features and columns correspond to batch.

Objects from the Class

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

Methods

- `batchNames` signature(object = "AssayData"): ...
- `batchNames<` signature(object = "AssayData"): ...
- `corr` signature(object = "AssayData", allele = "character"): ...
- `nu` signature(object = "AssayData", allele = "character"): ...
- `phi` signature(object = "AssayData", allele = "character"): ...

Details

- `lm`: Extracts entire list of linear model parameters.
- `corr`: The within-genotype correlation of log2(A) and log2(B) intensities.
- `nu`: The intercept for the linear model. The linear model is fit to the A and B alleles independently.
- `phi`: The slope for the linear model. The linear model is fit independently to the A and B alleles.

See Also

- CNSet-class

Examples

```r
library(crlmm)
library(Biocbase)
data(cnSetExample, package="crlmm")
cnSet <- cnSetExample
isCurrent(cnSet)
assayDataElementNames(batchStatistics(cnSet))
## Accessors for linear model parameters
## -- Included here primarily as a check that accessors are working
## -- Values are all NA until CN estimation is performed using the crlmm package
##
## subsetting
cnSet[1:10, ]
```
AssayDataList

## names of elements in the object
## accessors for parameters
nu(cnSet, "A")[1:10, ]
nu(cnSet, "B")[1:10, ]
phi(cnSet, "A")[1:10, ]
phi(cnSet, "B")[1:10, ]

## AssayDataList

### Create a list of assay data elements

#### Description

The eSetList-derived classes have an assayDataList slot instead of an assayData slot.

#### Usage

AssayDataList(storage.mode = c("lockedEnvironment", "environment", "list"), ...)

#### Arguments

- `storage.mode`: See assayDataNew.
- `...`: Named lists of matrices

#### Value

- `environment`

#### Author(s)

R.Scharpf

#### See Also

- assayDataNew

#### Examples

```r
r <- replicate(5, matrix(rnorm(25), 5, 5), simplify=FALSE)
r <- lapply(r, function(x,dns) {dimnames(x) <- dns; return(x)}, dns=list(letters[1:5], LETTERS[1:5]))
ad <- AssayDataList(r=r)
ls(ad)
```
batch

Accessor for slot assayDataList in Package oligoClasses

Description

Accessor for slot assayDataList in Package oligoClasses

Methods

signature(object = "gSetList") An object inheriting from class gSetList.
signature(object = "oligoSetList") An object inheriting from class gSetList.

batch

The batch variable for the samples.

Description

Copy number estimates are susceptible to systematic differences between groups of samples that were processed at different times or by different labs. While 'batch' is often unknown, a useful surrogates is often the scan date of the arrays (e.g., the month of the calendar year) or the 96 well chemistry plate on which the samples were arrayed during lab processing.

Usage

batch(object)
batchNames(object)
batchNames(object) <- value

Arguments

object
value

An object of class CNSet.
For 'batchNames', the value must be a character string corresponding of the unique batch names.

Value

The method 'batch' returns a character vector that has the same length as the number of samples in the CNSet object.

Author(s)

R. Scharpf

See Also

CNSet-class
Examples

```r
a <- matrix(1:25, 5, 5)
colnames(a) <- letters[1:5]
object <- new("CNSet", alleleA=a, batch=rep("batch1", 5))
batch(object)
batchNames(object)
```

Description

The `batchStatistics` slot contains statistics estimated from each batch that are used to derive copy number estimates.

Usage

```r
batchStatistics(object)
batchStatistics(object) <- value
```

Arguments

- `object` : An object of class `CNSet`
- `value` : An object of class `AssayData`

Details

An object of class `AssayData` for slot `batchStatistics` is initialized automatically when creating a new `CNSet` instance. Required in the call to `new` is a factor called `batch` whose unique values determine the number of columns for each assay data element.

Value

`batchStatistics` is an accessor for the slot `batchStatistics` that returns an object of class `AssayData`.

See Also

`CNSet-class`, `batchNames`, `batch`
BeadStudioSet-class  

Class "BeadStudioSet"

Description

A container for log R ratios and B allele frequencies from SNP arrays.

Objects from the Class

Objects can be created by calls of the form `new("BeadStudioSet", assayData, phenoData, featureData, experimentData, annotation, protocolData, baf, lrr, ...)`.

Slots

featureData: Object of class "GenomeAnnotatedDataFrame" ~~
assayData: Object of class "AssayData" ~~
phenoData: Object of class "AnnotatedDataFrame" ~~
experimentData: Object of class "MIAndE" ~~
annotation: Object of class "character" ~~
protocolData: Object of class "AnnotatedDataFrame" ~~
genome: Object of class "character" ~~
.baseClassVersion_: Object of class "Versions" ~~

Extends

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

Methods

In the methods below, object has class BeadStudioSet.

baf(object): accessor for the matrix of B allele frequencies.
baf(object) <- value: replacement method for B allele frequencies: value must be a matrix of integers.

as(object, "data.frame"): coerce to data.frame with column headers 'lrr', 'baf', 'x' (physical position with unit Mb), 'id', and 'is.snp'. Used for plotting with lattice.
copyNumber(object): accessor for log R ratios.
copyNumber(object) <- value: replacement method for the log R ratios
initialize signature(.Object = "BeadStudioSet"): constructs an instance of the class
lrr(object): accessor for matrix of log R ratios
lrr(object) <- value: replacement method for log R ratios: value should be a matrix or a ff_matrix.
show(object): print a short summary of the BeadStudioSet object.
updateObject(object): update a BeadStudioSet object.
**BeadStudioSetList-class**

**Author(s)**
R. Scharpf

**Examples**

```r
new("BeadStudioSet")
```

---

**BeadStudioSetList-class**

*List classes with assay data listed by chromosome*

**Description**

Container for log R ratios and B allele frequencies stored by chromosome.

**Slots**

- `assayDataList`: Object of class "AssayData"
- `phenodata`: Object of class "AnnotatedDataFrame"
- `featureDataList`: Object of class "list"
- `chromosome`: Object of class "integer"
- `annotation`: Object of class "character"
- `genome`: Object of class "character" indicating the genome build. Valid entries are "hg18" and "hg19".

**Methods defined for the class**

```r
clone2(object, id, prefix=",", ...)
```

Performs a deep copy of the ff objects in the assay data elements of object. A new object of the same class will be instantiated. The ff objects in the instantiated object will point to ff files on disk with prefix given by the argument `prefix`.

A use-case for such a function is that one may want to perform wave correction on the log R ratios in object, but keep a copy of the original unadjusted log R ratios. If object is not copied using `clone2` prior to wave correction, the log R ratios will be updated on disk and the original, unadjusted log R ratios will no longer be available.

**Accessors**

```r
baf(object)
```

An accessor for the B allele frequencies (BAFs). The accessor returns a list where each element of the list is a matrix of the BAFs for the corresponding element in the SetList object. While the BAFs have a range [0, 1], they are often saved internally as integers by multiplying the original BAFs by 1000. Users can restore the original scale by dividing by 1000.
lrr(object) An accessor for the log R ratios, an estimate of the copy number (presumably relative to diploid copy number) at each marker on a SNP array. The accessor returns a list where each element of the list is a matrix of the log R ratios for the corresponding element in the SetList object. The log R ratios are often saved internally as integers by multiplying the original LRRs by 100 in order to reduce the memory footprint of large studies. Users can restore the original scale by dividing by 100.

Author(s)

R. Scharpf

See Also

See supporting packages for methods defined for the class.

celfileDate

Description

Parses cel file dates from the header of .CEL files for the Affymetrix platform

Usage

celfileDate(filename)

Arguments

filename Name of cel file

Value

character string

Author(s)

H. Jaffee

Examples

require(hapmap.snp6)
path <- system.file("celFiles", package="hapmap.snp6")
celfiles <- list.celfiles(path, full.names=TRUE)
dts <- sapply(celfiles, celfileDate)
celfilename

*Extracts complete cel file name from a CNSet object*

**Description**

Returns the complete cel file (including path) for a CNSet object

**Usage**

celfilename(object)

**Arguments**

- **object**: An object of class CNSet

**Value**

Character string vector.

**Note**

If the CEL files for an experiment are relocated, the datadir should be updated accordingly. See examples.

**Author(s)**

R. Scharpf

**Examples**

```r
## Not run:
if(require(crlmm)){
  data(cnSetExample, package="crlmm")
  celfilename(cnSetExample)
}
## End(Not run)
```
checkExists  Checks to see whether an object exists and, if not, executes the appropriate function.

Description

Only loads an object if the object name is not in the global environment. If not in the global environment and the file exists, the object is loaded (by default). If the file does not exist, the function FUN is run.

Usage

checkExists(.name, .path = ".", .FUN, .FUN2, .save.it=TRUE, .load.it, ...)

Arguments

- .name  Character string giving name of object in global environment
- .path  Path to where the object is saved.
- .FUN   Function to be executed if <name> is not in the global environment and the file does not exist.
- .FUN2  Not currently used.
- .save.it Logical. Whether to save the object to the directory indicated by path. This argument is ignored if the object was loaded from file or already exists in the .GlobalEnv.
- .load.it Logical. If load.it is TRUE, we try to load the object from the indicated path. The returned object will replace the object in the .GlobalEnv unless the object is bound to a different name (symbol) when the function is executed.
- ...    Additional arguments passed to FUN.

Value

Could be anything – depends on what FUN, FUN2 perform.

Future versions could return a 0 or 1 indicating whether the function performed as expected.

Author(s)

R. Scharpf

Examples

path <- tempdir()
dir.create(path)
x <- 3+6
x <- checkExists("x", .path=path, .FUN=function(y, z) y+z, y=3, z=6)
rm(x)
x <- checkExists("x", .path=path, .FUN=function(y, z) y+z, y=3, z=6)
checkOrder

```
rm(x)
x <- checkExists("x", .path=path, .FUN=function(y, z) y+z, y=3, z=6)
rm(x)
## now there is a file called x.rda in tempdir(). The file will be loaded
x <- checkExists("x", .path=path, .FUN=function(y, z) y+z, y=3, z=6)
rm(x)
unlink(path, recursive=TRUE)
```

describe

### Description

Checks whether a eSet-derived class is ordered by chromosome and physical position

### Usage

```r
checkOrder(object, verbose = FALSE)
chromosomePositionOrder(object, ...)
```

### Arguments

- **object**  
  A SnpSet or CopyNumberSet.
- **verbose**  
  Logical.
- **...**  
  additional arguments to order

### Details

Checks whether the object is ordered by chromosome and physical position.

### Value

Logical

### Author(s)

R. Scharpf

### See Also

- order
Examples

data(oligoSetExample)
if(!checkOrder(oligoSet)){
  oligoSet <- chromosomePositionOrder(oligoSet)
}
checkOrder(oligoSet)

Methods

The methods for chromosome extracts the chromosome (represented as an integer) for each marker in an eSet-derived class or a AnnotatedDataFrame-derived class.

signature(object = "AnnotatedDataFrame") Accessor for chromosome.
signature(object = "eSet") If 'chromosome' is included in fvarLabels(object), the integer representation of the chromosome will be returned. Otherwise, an error is thrown.
signature(object = "GenomeAnnotatedDataFrame") Accessor for chromosome. If annotation was not available due to a missing or non-existent annotation package, the value returned by the accessor will be a vector of zero's.
(chromosome(object) <- value): Assign chromosome to the AnnotatedDataFrame slot of an eSet-derived object.
signature(object = "RangedDataCNV") Accessor for chromosome.

Note

Integer representation: chr X = 23, chr Y = 24, chr XY = 25. Symbols M, Mt, and MT are coded as 26.

See Also

chromosome2integer

Examples

chromosome2integer(c(1:22, "X", "Y", "XY", "M"))
chromosome2integer Converts chromosome to integer

Description

Coerces character string for chromosome in the pd. annotation packages to integers

Usage

```r
chromosome2integer(chrom)
integer2chromosome(intChrom)
```

Arguments

- `chrom`: A one or 2 letter character string (e.g., "1", "X", "Y", "MT", "XY")
- `intChrom`: An integer vector with values 1-25 possible

Details

This is useful when sorting SNPs in an object by chromosome and physical position – ensures that the sorting is done in the same way for different objects.

Value

- `integer2chromosome` returns a vector of character string indicating the chromosome the same length as `intChrom`.
- `chromosome2integer` returns a vector of integers the same length as the number of elements in the `chrom` vector.

Author(s)

R. Scharpf

Examples

```r
chromosome2integer(c(1:22, "X", "Y", "XY", "M"))
integer2chromosome(chromosome2integer(c(1:22, "X", "Y", "XY", "M")))
```
Description

CNSet is a container for intermediate data and parameters pertaining to allele-specific copy number estimation. Methods for CNSet objects, including accessors for linear model parameters and allele-specific copy number are included here.

Objects from the Class

An object from the class is not generally intended to be initialized by the user, but returned by the genotype function in the crlmm package.

The following creates a very basic CNSet with assayData containing the required elements.

```r
new(CNSet, alleleA=new("matrix"), alleleB=new("matrix"), call=new("matrix"), callProbability=new("n"))
```

Slots

- `batch`: Object of class "factor"
- `batchStatistics`: Object of class "AssayData"
- `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"
- `protocolData`: Object of class "AnnotatedDataFrame"
- `datadir`: Object of class "list"
- `mixtureParams`: Object of class "matrix"
- `__classVersion__`: Object of class "Versions"

Methods

The argument object for the following methods is a CNSet.

- `object[i, j]`: subset the CNSet object by markers (i) and/or samples (j).
- `A(object)`: accessor for the normalized intensities of allele A
- `A(object) <- value`: replace intensities for the A allele intensities by value. The object value must be a matrix, ff_matrix, or ffdf.
- `allele(object, allele)`: accessor for the normalized intensities for the A or B allele. The argument for allele must be either 'A' or 'B'
- `B(object)`: accessor for the normalized intensities of allele B
B(object) <- value: replace intensities for the B allele intensities by value. The object value must be a matrix, ff_matrix, or ffdf.
batch(object): vector of batch labels for each sample.
batchNames(object): the unique batch names
batchNames(object) <- value: relabel the batches
calls(object): accessor for genotype calls coded as 1 (AA), 2 (AB), or 3 (BB). Nonpolymorphic markers are NA.
confs(object): accessor for the genotype confidence scores.
close(object): close any open file connections to ff objects stored in the CNSet object.
as(object, "oligoSnpsSet"): coerce a CNSet object to an object of class oligoSnpsSet – a container for the total copy number and genotype calls.
corr(object): the correlation of the A and B intensities within each genotype.
flags(object): flags to indicate possible problems with the copy number estimation. Not fully implemented at this point.
new("CNSet"): instantiating a CNSet object.
nu(object, allele): accessor for the intercept (background) for the A and B alleles. The value of allele must be 'A' or 'B'.
on(object) open file connections for all ff objects stored in the CNSet object.
nu(object, allele): accessor for the slope for the A and B alleles. The value of allele must be 'A' or 'B'.
sigma2(object, allele): accessor for the within genotype variance
tau2(object, allele): accessor for background variance

Author(s)
R. Scharpf

Examples
new("CNSet")

CopyNumberSet-class  Class "CopyNumberSet"

Description
Container for storing total copy number estimates and confidence scores of the copy number estimates.

Objects from the Class
Objects can be created by calls of the form new("CopyNumberSet", assayData, phenoData, featureData, experimentData, annotationData, protocolData, copynumberData, cnconfidenceData, NNNIData).
Slots

- **assayData**: Object of class "AssayData"
- **phenodata**: Object of class "AnnotatedDataFrame"
- **featureData**: Object of class "AnnotatedDataFrame"
- **experimentData**: Object of class "MIAXE"
- **annotation**: Object of class "character"
- **protocolData**: Object of class "AnnotatedDataFrame"
- **.__classVersion__**: Object of class "Versions"

Extends

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

Methods

- **cnConfidence** signature(object = "CopyNumberSet"): ...
- **cnConfidence<-** signature(object = "CopyNumberSet", value = "matrix"): ...
- **coerce** signature(from = "CNSet", to = "CopyNumberSet"): ...
- **copyNumber** signature(object = "CopyNumberSet"): ...
- **copyNumber<-** signature(object = "CopyNumberSet", value = "matrix"): ...
- **initialize** signature(Object = "CopyNumberSet"): ...

Note

This container is primarily for platforms for which genotypes are unavailable. As oligoSnpSet extends this class, methods related to total copy number that do not depend on genotypes can be defined at this level.

Author(s)

R. Scharpf

See Also

For genotyping platforms, total copy number estimates and genotype calls can be stored in the oligoSnpSet class.

Examples

```r
showClass("CopyNumberSet")
cnset <- new("CopyNumberSet")
ls(Biobase::assayData(cnset))
```
CopyNumberSet-methods

Methods for class CopyNumberSet.

Description

Accessors and CopyNumberSet

Usage

copyNumber(object, ...)
cnConfidence(object)
copyNumber(object) <- value
cnConfidence(object) <- value

Arguments

object CopyNumberSet object or derived class
... Ignored for CopyNumberSet and oligoSnSet.
value matrix

Value

copyNumber returns a matrix of copy number estimates or relative copy number estimates. Since the copy number estimates are stored as integers (copy number * 100), the matrix returned by the copyNumber accessor will need to be divided by a factor of 100 to transform the measurements back to the original copy number scale.

cnConfidence returns a matrix of confidence scores for the copy number estimates. These are also represented as integers and will require a back-transformation to the original scale.

Examples

library(Biobase)
data(oligoLevelData)
path <- system.file("extdata", package="oligoClasses")
fd <- readRDS(file.path(path, "genomeAnnotatedDataFrameExample.rds"))
## the following command creates an 'oligoSnSet' object, storing
## an integer representation of the log2 copy number in the 'copyNumber' element
## of the assayData. Genotype calls and genotype confidence scores are also stored
## in the assayData.
oligoSet <- new("oligoSnSet",
copyNumber=integerMatrix(log2(oligoLevelData["copynumber"])/100, 100),
call=oligoLevelData["genotypes"],
callProbability=integerMatrix(oligoLevelData["callmmConfidence"], 1),
annotation=oligoLevelData["platform"],
featureData=fd,
genome="hg19")
## createFF

Create ff objects.

### Description

Creates ff objects (array-like) using settings (path) defined by oligoClasses.

### Usage

```r
createFF(name, dim, vmode = "double", initdata = NULL)
```

### Arguments

- **name**: Prefix for filename.
- **dim**: Dimensions.
- **vmode**: Mode.
- **initdata**: NULL.

### Value

- ff object.
Note
This function is meant to be used by developers.

See Also
ff

---

**db**  
*Get the connection to the SQLite Database*

**Description**
This function will return the SQLite connection to the database associated to objects used in oligo.

**Usage**

db(object)

**Arguments**

object Object of valid class. See methods.

**Value**
SQLite connection.

**Methods**

object = "FeatureSet" object of class FeatureSet  
object = "SnpCallSet" object of class SnpCallSet  
object = "DBPDInfo" object of class DBPDInfo  
object = "SnpLevelSet" object of class SnpLevelSet

**Author(s)**
Benilton Carvalho

**Examples**

## db(object)
**Class "DBPDInfo"**

**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)

**Methods**
- `annotation`: string describing annotation package associated to object

**ExpressionFeatureSet Object**

**Description**
Example of ExpressionFeatureSet Object.

**Usage**
```
data(efsExample)
```

**Format**
Object belongs to ExpressionFeatureSet class.

**Examples**
```
data(efsExample)
class(efsExample)
```
**exprs-methods**

*Accessor for the 'exprs' slot*

**Description**

Accessor for the 'exprs'/se.exprs' slot of FeatureSet-like objects

**Methods**

- **object = "ExpressionSet"** Expression matrix for objects of this class. Usually results of preprocessing algorithms, like RMA.
- **object = "FeatureSet"** General container 'exprs' inherited from eSet
- **object = "SnpSet"** General container 'exprs' inherited from eSet, not yet used.

**featureDataList-methods**

*Accessor for slot featureDataList in Package oligoClasses ~*

**Description**

Accessor for slot featureDataList in Package oligoClasses ~

**Methods**

- **signature(object = "gSetList")** An object inheriting from class gSetList.

**FeatureSet-class**

"FeatureSet" and "FeatureSet" Extensions

**Description**

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

**Objects from the Class**

The FeatureSet class is VIRTUAL. Therefore users are not able to create instances of such class.

Objects for FeatureSet-like classes can be created by calls of the form: new(CLASSNAME, assayData, manufacturer, platform, exprs, phenodata, featuredata, experimentdata, annotation). However, the preferred way is using parsers like read.celfiles and read.xysfiles.
Slots

manufacturer: 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"

Methods

show signature(.Object = "FeatureSet"): show object contents
bothStrands signature(.Object = "SnpFeatureSet"): checks if object contains data for both strands simultaneously (50K/250K Affymetrix SNP chips - in this case it returns TRUE); if object contains data for one strand at a time (SNP 5.0 and SNP 6.0 - in this case it returns FALSE)

Author(s)

Benilton Carvalho

See Also

eSet, VersionedBiobase, Versioned

Examples

set.seed(1)
tmp <- 2*rnorm(100), ncol=4)
rownames(tmp) <- 1:25
colnames(tmp) <- paste("sample", 1:4, sep="")
efs <- new("ExpressionFeatureSet", exprs=tmp)

---

ffdf-class

Class ‘ffdf’

Description

Extended package ff’s class definitions for ff to S4.

Objects from the Class

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

.S3Class: Object of class ffdf

Extends

Class "oldClass", directly. Class "list_or_ffdf", directly.

Methods

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

Class "ff_matrix"

Description

~~ A concise (1-5 lines) description of what the class is. ~~

Objects from the Class

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

Slots

.S3Class: Object of class "character"

Extends

Class "oldClass", directly.

Methods

annotatedDataFrameFrom signature(object = "ff_matrix"): ...

Examples

showClass("ff_matrix")
**Description**

A class union of 'ffdf', 'ff_matrix', and 'matrix'

**Objects from the Class**

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

**Methods**

`GenomeAnnotatedDataFrameFrom` signature(object = "ff_or_matrix"): ...

**Author(s)**

R. Scharpf

**See Also**

`ff, ffdf`

**Examples**

```r
showClass("ff_or_matrix")
```

---

**fileConnections**

Open and close methods for matrices and numeric vectors

**Description**

CNSet objects can contain ff-derived objects that contain pointers to files on disk, or ordinary matrices. Here we define open and close methods for ordinary matrices and vectors that that simply pass back the original matrix/vector.

**Usage**

```r
open(con, ...)  
openff(object)  
closeff(object)
```

**Arguments**

- `con` matrix or vector
- `object` A CNSet object.
- `...` Ignored
flags

Value
not applicable

Author(s)
R. Scharpf

Examples
open(rnorm(15))
open(matrix(rnorm(15), 5, 3))

flags
Batch-level summary of SNP flags.

Description
Used to flag SNPs with low minor allele frequencies, or for possible problems during the CN estimation step. Currently, this is primarily more for internal use.

Usage
flags(object)

Arguments
object An object of class CNSet

Value
A matrix or ff_matrix object with rows corresponding to markers and columns corresponding to batch.

See Also
batchStatistics

Examples
x <- matrix(runif(250*96*2, 0, 2), 250, 96*2)
test1 <- new("CNSet", alleleA=x, alleleB=x, call=x, callProbability=x,
batch=as.character(rep(letters[1:2], each=96)))
dim(flags(test1))
**Description**

Miscellaneous generics. Methods defined in packages that depend on oligoClasses

**Usage**

baf(object)

lrr(object)

**Arguments**

object A eSet-derived class.

**Author(s)**

R. Scharpf

---

**Description**

AnnotatedDataFrame with genomic coordinates (chromosome, position)

**Slots**

varMetadata: Object of class "data.frame"

data: Object of class "data.frame"

dimLabels: Object of class "character"

.__classVersion__: Object of class "Versions"

**Extends**

Class "AnnotatedDataFrame", directly. Class "Versioned", by class "AnnotatedDataFrame", distance 2.
GenomeAnnotatedDataFrame-class

Coercion to or from other classes

as(from, "GenomeAnnotatedDataFrame"): 
Coerce an object of class AnnotatedDataFrame to a GenomeAnnotatedDataFrame.

makeFeatureGRanges(object, genome, ...):
Construct a GRanges instance from a GenomeAnnotatedDataFrame object. genome is a character string indicating the UCSC build. Supported builds are "hg18" and "hg19", but are platform specific. In particular, some platforms only support build hg19 at this time.

updateObject(object): 
For updating a GenomeAnnotatedDataFrame

Accessors

chromosome(object), chromosome(object) <- value 
Get or set chromosome.

isSnp(object):
Many platforms include polymorphic and nonpolymorphic markers. isSnp evaluates to TRUE if the marker is polymorphic.

position(object):
Physical position in the genome

getArm(object, genome):
Retrieve character vector indicating the chromosome arm of each marker in object. genome should indicate which genome build was used to define the chromosomal locations (currently, only UCSC genome builds 'hg18' and 'hg19' supported for this function).

Author(s)

R. Scharpf

See Also

AnnotatedDataFrame

Examples

new("GenomeAnnotatedDataFrame") 
if(require("pd.mapping50k.hind240") && require("pd.mapping50k.xba240") && require("SNPchip")){ 
data(locusLevelData) 
gd <- GenomeAnnotatedDataFrameFrom(locusLevelData[["genotypes"]], 
annotationPkg=locusLevelData[["platform"]], 
genome="hg19") 
arm <- getArm(gd, "hg19")
}
Description

GenomeAnnotatedDataFrameFrom is a convenience for creating GenomeAnnotatedDataFrame objects.

Methods

Use the method with GenomeAnnotatedDataFrameFrom(object, annotationPkg, genome, ...); the argument annotationPkg must be specified for matrix and AssayData classes.

signature(object="assayData") This method creates an GenomeAnnotatedDataFrame using feature names and dimensions of an AssayData object as a template.

signature(object="matrix") This method creates an GenomeAnnotatedDataFrame using row names and dimensions of a matrix object as a template.

signature(object="NULL") This method (called with 'NULL' as the object) creates an empty GenomeAnnotatedDataFrame.

signature(object="array") This method (called with 'array' as the object) creates a GenomeAnnotatedDataFrame using the first dimension of the array (rows are the number of features).

Author(s)

R Scharpf

Examples

```r
require(Biobase)
minReqVersion <- "1.0.2"
require(human370v1cCrlmm)
if (packageDescription("human370v1cCrlmm", fields='Version') >= minReqVersion){
x <- matrix(1:25, 5, 5,
   dimnames=list(c("rs10000092","rs10000855","rs100016","rs10003241","rs10004197"), NULL))
gd <- GenomeAnnotatedDataFrameFrom(x, annotationPkg="human370v1cCrlmm",
genome="hg18")
pData(gd)
chromosome(gd)
position(gd)
}
```
**genomeBuild**

---

**Genome Build Information**

**Description**

Returns the genome build. This information comes from the annotation package and is given as an argument during the package creation process.

**Usage**

```r
geneBuild(object)
```

**Arguments**

- **object**: Supported objects include `PDInfo`, `FeatureSet`, and any `gSet`-derived or `eSetList`-derived object.

**Value**

character string

**Note**

Supported builds are UCSC genome builds are 'hg18' and 'hg19'.

**Examples**

```r
showMethods("geneBuild", where="package:oligoClasses")
```

---

**geometry**

---

**Array Geometry Information**

**Description**

For a given array, `geometry` returns the physical geometry of it.

**Usage**

```r
geometry(object)
```

**Arguments**

- **object**: `PDInfo` or `FeatureSet` object

**Examples**

```r
if (require(pd.mapping50k.xba240))
geometry(pd.mapping50k.xba240)
```
Description

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

Usage

geta(object)
gatem(object)
A(object, ...)
B(object, ...)

Arguments

object: SnpQSet, SnpCnvQSet or TilingFeatureSet2 object.
... arguments to be passed to allele - 'sense' and 'antisense' are valid values if the array is pre-SNP_5.0

Details

For SNP data, SNPRMA summarizes the SNP information into 4 quantities (log2-scale):
- antisenseThetaAantisense allele A. (Not applicable for Affymetrix 5.0 and 6.0 platforms.)
- antisenseThetaBantisense allele B. (Not applicable for Affymetrix 5.0 and 6.0 platforms.)
- senseThetaAsense allele A. (Not applicable for Affymetrix 5.0 and 6.0 platforms.)
- senseThetabSense allele B. (Not applicable for Affymetrix 5.0 and 6.0 platforms.)
- alleleAAffymetrix 5.0 and 6.0 platforms
- alleleBAffymetrix 5.0 and 6.0 platforms

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.

For Tiling data, getM and getA return the log-ratio and average log-intensities computed across channels:
M = log2(channel1)-log2(channel2) A = (log2(channel1)+log2(channel2))/2

When large data support is enabled with the ff package, the AssayData elements of an AlleleSet object can be ff_matrix or ffdf, in which case pointers to the ff object are stored in the assay data.
The functions open and close can be used to open or close the connection, respectively.

Value

A 3-dimensional array (SNP's x Samples x Strand) with the requested measure, when the input SNP data (50K, 250K).
A 2-dimensional array (SNP's x Samples), when the input is from SNP 5.0 and SNP 6.0 arrays.
A 2-dimensional array if the input is from Tiling arrays.
getBar

See Also
    snprma

getBar

Gets a bar of a given length.

Description
    Gets a bar of a given length.

Usage
    getBar(width = getOption("width"))

Arguments
    width    desired length of the bar.

Value
    character string.

Author(s)
    Benilton S Carvalho

Examples
    message(getBar())

getSequenceLengths

Load chromosome sequence lengths for UCSC genome build hg18 or hg19

Description
    Load chromosome sequence lengths for UCSC genome build hg18 or hg19

Usage
    getSequenceLengths(build)

Arguments
    build    character string: "hg18" or "hg19"
Details

The chromosome sequence lengths for UCSC builds hg18 and hg19 were extracted from the packages BSgenome.Hsapiens.UCSC.hg18 and BSgenome.Hsapiens.UCSC.hg19, respectively.

Value

Names integer vector of chromosome lengths.

Author(s)

R. Scharpf

Examples

getSequenceLengths("hg18")
getSequenceLengths("hg19")

if(require("GenomicRanges")){
  ## from GenomicRanges
  sl <- getSequenceLengths("hg18")[[c("chr1", "chr2", "chr3")]]
  gr <- GRanges(seqnames =
                Rle(c("chr1", "chr2", "chr1", "chr3"), c(1, 3, 2, 4)),
                ranges =
                IRanges(1:10, width = 10:1, names = head(letters,10)),
                strand =
                Rle(strand(c("-", "+", "x", "+", "-")),
                      c(1, 2, 2, 3, 2)),
                score = 1:10,
                GC = seq(1, 0, length=10),
                seqlengths=sl)
  metadata(gr) <- list(genome="hg18")
  gr
  metadata(gr)
}

Description

Methods for GRanges objects
**findOverlaps methods**

`findOverlaps(query, subject, ...)`:
Find the feature indices in `subject` that overlap the genomic intervals in `query`, where `query` is a GRanges object and `subject` is a gSet-derived object. Additional arguments to the `findOverlaps` method in the package `IRanges` can be passed through the `...` operator.

**Accessors**

- `object` is an instance of the GRanges class.
  - `coverage2(object)`:
    For the GRanges and GRangesList objects returned by the hidden Markov model implemented in the "VanillaICE" package and the segmentation algorithm in the "MinimumDistance" package, the intervals are annotated by the number of probes (markers) for SNPs and nonpolymorphic regions. coverage2 and numberProbes are convenient accessors for these annotations.
  - `genomeBuild(object)`:
    Accessor for the UCSC genome build.
  - `numberProbes(object)`:
    Integer vector indicating the number of probes (markers) for each range in `object`. Equivalent to coverage2.
  - `state(object)`:
    Accessor for the elementMetadata column 'state', when applicable. State is used to contain the index of the inferred copy number state for various hmm methods defined in the VanillaICE.

**See Also**

GRanges

**Examples**

```r
library(IRanges)
library(GenomicRanges)
gr1 <- GRanges(seqnames = "chr2", ranges = IRanges(3, 6),
state=3L, numberProbes=100L)
## convenience functions
state(gr1)
numberProbes(gr1)

g2 <- GRanges(seqnames = c("chr1", "chr1"),
ranges = IRanges(c(7,13), width = 3),
state=c(2L, 2L), numberProbes=c(200L, 250L))
g3 <- GRanges(seqnames = c("chr1", "chr2"),
ranges = IRanges(c(1, 4), c(3, 9),
state=c(1L, 4L), numberProbes=c(300L, 350L))
## Ranges organized by sample
g1 <- GRangesList("sample1" = gr1, "sample2" = gr2, "sample3" = gr3)
sampleNames(gr1) ## same as names(gr1)
numberProbes(gr1)
chromosome(gr1)
```
gSet-class

Container for objects with genomic annotation on SNPs

Description

Container for objects with genomic annotation on SNPs

Objects from the Class

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

Slots

featureData: Object of class "GenomeAnnotatedDataFrame"
assayData: Object of class "AssayData"
phenoData: Object of class "AnnotatedDataFrame"
experimentData: Object of class "MIAxE"
annotation: Object of class "character"
protocolData: Object of class "AnnotatedDataFrame"
genome: 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

The object for the below methods is a class that extends the virtual class gSet.

checkOrder(object): checks that the object is ordered by chromosome and physical position.
   Returns logical.

chromosome(object): accessor for chromosome in the GenomeAnnotatedDataFrame slot.

chromosome(object) <- value: replacement method for chromosome in the GenomeAnnotatedDataFrame slot. value must be an integer vector.

db(object): database connection
genomeBuild(object) <- value:
   Get or set the UCSC genome build. Supported builds are hg18 and hg19.
getArm(object): Character vector indicating the chromosomal arm for each marker in object.
isSnp(object): whether the marker is polymorphic. Returns a logical vector.
makeFeatureGRanges(object): Construct an instance of the GRanges class from a GenomeAnnotatedDataFrame.
position(object): integer vector of the genomic position
show(object):
   Print a concise summary of object.

Author(s)
   R. Scharpf

See Also
   chromosome, position, isSnp

Examples
   showClass("gSet")

---

gSetList-class  Virtual Class for Lists of eSets

Description
   Virtual Class for Lists of eSets.

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

Slots
   assayDataList: Object of class "AssayData" ~
   phenoData: Object of class "AnnotatedDataFrame" ~
   protocolData: Object of class "AnnotatedDataFrame" ~
   experimentData: Object of class "MIAME" ~
   featureDataList: Object of class "list" ~
   chromosome: Object of class "vector" ~
   annotation: Object of class "character" ~
   genome: Object of class "character" ~
Accessors

object is an instance of a gSetList-derived class.

annotation(object):
    character string indicating the package used to provide annotation for the features on the array.

chromosome(object):
    Returns the chromosome corresponding to each element in the gSetList object.

elementLengths(object): Returns the number of rows for each list of assays. In most gSetList-derived classes, the assays are organized by chromosome and elementLengths returns the number of markers for each chromosome.

genomeBuild(object), genomeBuild(object) <- value:
    Get or set the UCSC genome build. Supported builds are hg18 and hg19.

Coercion

object is an instance of a gSetList-derived class.

makeFeatureGRanges(object, ...):
    Create a GRanges object for the featureData. The featureData is stored as a list. This method stacks the featureData from each list element. Metadata columns in the GRanges object include physical position (’position’), a SNP indicator (’isSnp’), and the chromosome. The genome build is extracted from object using the method genomeBuild.

Author(s)

R. Scharpf

See Also

oligoSetList, BeadStudioSetList

Examples

showClass("gSetList")

Description

Functions to convert probabilities to integers, or integers to probabilities.

Probabilities estimated in the crlmm package are often stored as integers to save memory. We provide a few utility functions to go back and forth between the probability and integer representations.
initializeBigMatrix

Usage

i2p(i)
p2i(p)

Arguments

i A matrix or vector of integers.
p A matrix or vector of probabilities.

Value

The value returned by i2p is
1 - exp(-i/1000)
The value returned by p2i is
as.integer(-1000*log(1-p))

See Also

cfabs

Examples

i2p(693)
p2i(0.5)
i2p(p2i(0.5))

initializeBigMatrix Initialize big matrices/vectors.

Description

Initialize big matrices or vectors appropriately (conditioned on the status of support for large datasets - see Details).

Usage

initializeBigMatrix(name=basename(tempfile()), nr=0L, nc=0L, vmode = "integer", initdata = NA)
initializeBigVector(name=basename(tempfile()), n=0L, vmode = "integer",
    initdata = NA)
initializeBigArray(name=basename(tempfile()), dim=c(0L,0L,0L),
    vmode="integer", initdata=NA)
Arguments

- **name**: prefix to be used for file stored on disk
- **nr**: number of rows
- **nc**: number of columns
- **n**: length of the vector
- **vmode**: mode - "integer", "double"
- **initdata**: Default is NA
- **dim**: Integer vector indicating the dimensions of the array to initialize

Details

These functions are meant to be used by developers. They provide means to appropriately create big vectors or matrices for packages like oligo and crlmm (and friends). These objects are created conditioned on the status of support for large datasets.

Value

If the 'ff' package is loaded (in the search path), then an 'ff' object is returned. A regular R vector or array is returned otherwise.

Examples

```r
x <- initializeBigVector("test", 10)
class(x)
x
if (isPackageLoaded("ff"))
  finalizer(x) <- "delete"
rm(x)
initializeBigMatrix(nr=5L, nc=5L)
initializeBigArray(dim=c(10, 5, 3))
```

integerMatrix

Coerce numeric matrix (or array) to a matrix (array) of integers, retaining dimnames.

Description

Coerce numeric matrix to matrix of integers, retaining dimnames.

Usage

```r
integerMatrix(x, scale = 100)
integerArray(x, scale=100)
```
is.ffmatrix

Arguments

  x  a matrix or array

  scale scalar (numeric). If not 1, x is multiplied by scale prior to coercing to a matrix of integers.

Value

  A matrix or array of integers.

Author(s)

  R. Scharpf

Examples

  x <- matrix(rnorm(10), 5, 2)
  rownames(x) = letters[1:5]
  i <- integerMatrix(x, scale=100)

Description

  Check if object is an ff-matrix object.

Usage

  is.ffmatrix(object)

Arguments

  object object to be checked

Value

  Logical.

Note

  This function is meant to be used by developers.
isPackageLoaded

Description

Checks if package is loaded.

Usage

isPackageLoaded(pkg)

Arguments

pkg Package to be checked.

Details

Checks if package name is in the search path.

Value

Logical.

See Also

search

Examples

isPackageLoaded("oligoClasses")
isPackageLoaded("ff")
isPackageLoaded("snow")
Description

~~ Methods for function isSnp in package oligoClasses ~~

Methods

Return an indicator for whether the marker is polymorphic (value 1) or nonpolymorphic (value 0).

Return an indicator for whether the vector of marker identifiers in object is polymorphic. pkgname must be one of the supported annotation packages specific to the platform.

signature(object = "character", pkgname = "character")
signature(object = "eSet", pkgname = "ANY")
If 'isSnp' is included in fvarLabels(object), an indicator for polymorphic markers is returned. Otherwise, an error is thrown.

signature(object = "GenomeAnnotatedDataFrame", pkgname = "ANY") Accessor for indicator of whether the marker is polymorphic. If annotation was not available due to a missing or non-existent annotation package, the value returned by the accessor will be a vector of zero's.

kind

Array type

Description

Retrieves the array type.

Usage

kind(object)

Arguments

object FeatureSet or DBPDInfo object

Value

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

Examples

if (require(pd.mapping50k.xba240)){
  data(sfsExample)
  Biobase::annotation(sfsExample) <- "pd.mapping50k.xba240"
  kind(sfsExample)
}
ldSetOptions

Set/check large dataset options.

Description

Set/check large dataset options.

Usage

```r
ldSetOptions(nsamples=100, nprobesets=20000, path=getwd(), verbose=FALSE)
ldStatus(verbosity=FALSE)
ldPath(path)
```

Arguments

- `nsamples`: number of samples to be processed at once.
- `nprobesets`: number of probesets to be processed at once.
- `path`: path where to store large dataset objects.
- `verbose`: verbosity (logical).

Details

Some functions in oligo/crlmm can process data in batches to minimize memory footprint. When using this feature, the ‘ff’ package resources are used (and possibly combined with cluster resources set in options() via ‘snow’ package).

Methods that are executed on a sample-by-sample manner can use ocSamples() to automatically define how many samples are processed at once (on a compute node). Similarly, methods applied to probesets can use ocProbesets(). Users should set these options appropriately.

- `ldStatus` checks the support for large datasets.
- `ldPath` checks where ff files are stored.

Author(s)

Benilton S Carvalho

See Also

ocSamples, ocProbesets

Examples

```r
ldStatus(TRUE)
```
**length-methods**

Number of samples for FeatureSet-like objects.

**Description**
Number of samples for FeatureSet-like objects.

**Methods**

\[x = "\text{FeatureSet}"\] Number of samples

---

**library2**

Suppress package startup messages when loading a library

**Description**
Suppress package startup messages when loading a library

**Usage**

`library2(...)`

**Arguments**

... arguments to `library`

**Author(s)**

R. Scharpf

**See Also**

`library`

**Examples**

`library2("Biobase")`
list.celfiles  

List CEL files.

Description

Function used to get a list of CEL files.

Usage

list.celfiles(..., listGzipped=FALSE)

Arguments

... Passed to list.files
listGzipped Logical. List .CEL.gz files?

Value

Character vector with filenames.

Note

Quite often users want to use this function to pass filenames to other methods. In this situations, it is safer to use the argument 'full.names=TRUE'.

See Also

list.files

Examples

if (require(hapmapsnp5)){
  path <- system.file("celFiles", package="hapmapsnp5")

  ## only the filenames
  list.celfiles(path)

  ## the filenames with full path...
  ## very useful when genotyping samples not in the working directory
  list.celfiles(path, full.names=TRUE)
} else{
  ## this won't return anything
  ## if in the working directory there isn't any CEL
  list.celfiles(getwd())
}
ListClasses

\textit{eSetList class}

\textbf{Description}

Initialization method for \textit{eSetList} virtual class.

\textbf{locusLevelData \hspace{1cm} Basic data elements required for the HMM}

\textbf{Description}

This object is a list containing the basic data elements required for the HMM

\textbf{Usage}

data(locusLevelData)

\textbf{Format}

A list

\textbf{Details}

The basic assay data elements that can be used for fitting the HMM are:
1. a mapping of platform identifiers to chromosome and physical position
2. (optional) a matrix of copy number estimates
3. (optional) a matrix of confidence scores for the copy number estimates (e.g., inverse standard deviations)
4. (optional) a matrix of genotype calls
5. (optional) CRLMM confidence scores for the genotype calls
At least (2) or (4) is required. The \texttt{locusLevelData} is a list that contains (1), (2), (4), and (5).

\textbf{Source}

A HapMap sample on the Affymetrix 50k platform. Chromosomal alterations were simulated. The last 100 SNPs on chromosome 2 are, in fact, a repeat of the first 100 SNPs on chromosome 1 – this was added for internal use.

\textbf{Examples}

data(locusLevelData)
str(locusLevelData)
Construct a GRanges object from several possible feature-level classes

Description

Construct a GRanges object from several possible feature-level classes. The conversion is useful for subsequent ranged-data queries, such as findOverlaps, countOverlaps, etc.

Usage

makeFeatureGRanges(object, ...)

Arguments

object

A gSet-derived object containing chromosome and physical position for the markers on the array.

... See the makeFeatureGRanges method for GenomeAnnotatedDataFrame.

Value

A GRanges object.

Author(s)

R. Scharpf

See Also

findOverlaps, GRanges, GenomeAnnotatedDataFrame

Examples

if(require("VanillaICE")){
library(oligoClasses)
library(GenomicRanges)
library(Biobase)
library(foreach)
registerDoSEQ()
data(oligoSetExample, package="oligoClasses")
oligoSet <- oligoSet[chromosome(oligoSet) == 1, ]
grl <- hmm(oligoSet, TAUP=1e10)
class(grl)# GRangesList
gr <- grl[1]
(frange <- makeFeatureGRanges(oligoSet))
## which features overlap with the second range in sample NA06993
subsetByOverlaps(frange, gr[2,])
}
Description

Manufacturer ID for FeatureSet-like and DBPDInfo-like objects.

Methods

object = "FeatureSet"  Manufacturer ID
object = "PDInfo"  Manufacturer ID

oclapply  

_-lapply-like function that parallelizes code when possible._-

Description

oclapply is an _lapply-like_ function that checks if ff/snow are loaded and if the cluster variable is set to execute FUN on a cluster. If these requirements are not available, then _lapply_ is used.

Usage

oclapply(x, FUN, ..., neededPkgs)

Arguments

- **x**  _first argument to FUN._
- **FUN**  _function to be executed._
- **...**  _additional arguments to FUN._
- **neededPkgs**  _packages needed to execute FUN on the compute nodes._

Details

neededPkgs is needed when parallel computing is expected to be used. These packages are loaded on the compute nodes before the execution of FUN.

Value

A list of length length(X).

Author(s)

Benilton S Carvalho

See Also

_lapply, parStatus_
**Description**

Tools to simplify management of clusters via 'snow' package and large dataset handling through the 'bigmemory' package.

**Usage**

\[
\text{ocSamples}(n) \\
\text{ocProbesets}(n)
\]

**Arguments**

\[
n \quad \text{integer representing the maximum number of samples/probesets to be processed simultaneously on a compute node.}
\]

**Details**

Some methods in the oligo/crlmm packages, like backgroundCorrect, normalize, summarize and rma can use a cluster (set through the 'foreach' package). The use of cluster features is conditioned on the availability of the 'ff' (used to provide shared objects across compute nodes) and 'foreach' packages.

To use a cluster, 'oligo/crlmm' checks for three requirements: 1) 'ff' is loaded; 2) an adaptor for the parallel backend (like 'doMPI', 'doSNOW', 'doMC') is loaded and registered.

If only the 'ff' package is available and loaded (in addition to the caller package - 'oligo' or 'crlmm'), these methods will allow the user to analyze datasets that would not fit in RAM at the expense of performance.

In the situations above (large datasets and cluster), oligo/crlmm uses the options ocSamples and ocProbesets to limit the amount of RAM used by the machine(s). For example, if ocSamples is set to 100, steps like background correction and normalization process (in RAM) 100 samples simultaneously on each compute node. If ocProbesets is set to 10K, then summarization processes 10K probesets at a time on each machine.

**Warning**

In both scenarios (large dataset and/or cluster use), there is a penalty in performance because data are written to disk (to either minimize memory footprint or share data across compute nodes).

**Author(s)**

Benilton Carvalho
oligoSet

Examples

if(require(doMC)) {
  registerDoMC()
  ## tasks like summarize()
}

oligoSet

An example instance of oligoSnpSet class

Description

An example instance of the oligoSnpSet class

Usage

data(oligoSetExample)

Source

Created from the simulated locusLevelData provided in this package.

See Also

locusLevelData

Examples

## Not run:
## 'oligoSetExample' created by the following
data( locusLevelData )
oligoSet <- new( "oligoSnpSet",
copyNumber = integerMatrix( log2( locusLevelData[[ "copynumber" ]] / 100 ), 100 ),
call = locusLevelData[[ "genotypes" ]],
callProbability = locusLevelData[[ "crlmmConfidence" ]],
annotation = locusLevelData[[ "platform" ]],
genome = "hg19"
oligoSet <- oligoSet ! [ ! is.na( chromosome( oligoSet ) ) , ]
oligoSet <- oligoSet [ chromosome( oligoSet ) < 3 , ]

## End(Not run)
data( oligoSetExample )
oligoSet
Description

Methods for oligoSnpsSet class

Methods

In the following code, object is an instance of the oligoSnpsSet class.

\[
\text{new("oligoSnpsSet", ...)}: \text{Instantiates an object of class oligoSnpsSet. The assayData elements of the oligoSnpsSet class can include matrices of genotype calls, confidence scores for the genotype calls, B allele frequencies, absolute or relative copy number, and confidence scores for the copy number estimates. Each matrix should be coerced to an integer scale prior to assignment to the oligoSnpsSet object. Validity methods defined for the class will fail if the matrices are not integers. See examples for additional details.}
\]

\[
\text{baf(object): Accessor for integer representation of the B allele frequencies. The value returned by this method can be divided by 1000 to obtain B allele frequencies on the original [0,1] scale.}
\]

\[
\text{baf(object) <- value: Assign an integer representation of the B allele frequencies to the 'baf' element of the assayData slot. value must be a matrix of integers. See the examples for help converting BAFs to a matrix of integers.}
\]

Description

Checks if oligo/crlmm can use parallel resources.

Usage

parStatus()

Value

logical

Author(s)

Benilton S Carvalho
pdPkgFromBioC

Get packages from BioConductor.

Description

This function checks if a given package is available on BioConductor and installs it, in case it is.

Usage

pdPkgFromBioC(pkgname, lib = .libPaths()[1], verbose = TRUE)

Arguments

pkgname character. Name of the package to be installed.
lib character. Path where to install the package at.
verbose logical. Verbosity flag.

Details

Internet connection required.

Value

Logical: TRUE if package was found, downloaded and installed; FALSE otherwise.

Author(s)

Benilton Carvalho

See Also

download.packages

Examples

## Not run:
pdPkgFromBioC("pd.mapping50k.xba240")

## End(Not run)
platform-methods

Platform Information

Description
Platform Information

Methods

object = "FeatureSet"  platform information

pmFragmentLength-methods

Information on Fragment Length

Description
This method will return the fragment length for PM probes.

Methods

object = "AffySNPPDInfo"  On AffySNPPDInfo objects, it will return the fragment length that contains the SNP in question.

position-methods

Methods for function position in Package oligoClasses

Description
Methods for function position in package oligoClasses

Methods

The methods for position extracts the physical position stored as an integer for each marker in a eSet-derived class or a AnnotatedDataFrame-derived class.

signature(object = "AnnotatedDataFrame")  Accessor for physical position.

signature(object = "eSet")  If 'position' is included in fvarLabels(object), the physical position will be returned. Otherwise, an error is thrown.

signature(object = "GenomeAnnotatedDataFrame")  Accessor for physical position. If annotation was not available due to a missing or non-existent annotation package, the value returned by the accessor will be a vector of zero’s.
requireAnnotation

Helper function to load packages.

Description

This function checks the existence of a given package and loads it if available. If the package is not available, the function checks its availability on BioConductor, downloads it and installs it.

Usage

requireAnnotation(pkgname, lib=.libPaths()[1], verbose = TRUE)

Arguments

pkgname character. Package name (usually an annotation package).
lib character. Path where to install packages at.
verbose logical. Verbosity flag.

Value

Logical: TRUE if package is available or FALSE if package unavailable for download.

Author(s)

Benilton Carvalho

See Also

install.packages

Examples

## Not run:
requirePackage("pd.mapping50k.xba240")

## End(Not run)
requireClusterPkgSet  

DEPRECATED FUNCTIONS. Package loaders for clusters.

Description

Package loaders for clusters.

Usage

requireClusterPkgSet(packages)
requireClusterPkg(pkg, character.only)

Arguments

packages  
character vector with the names of the packages to be loaded on the compute nodes.

pkg  
name of a package given as a name or literal character string

character.only  
a logical indicating whether 'pkg' can be assumed to be a character string

Details

requireClusterPkgSet applies require for a set of packages on the cluster nodes.
requireClusterPkg applies require for *ONE* package on the cluster nodes and accepts every argument taken by require.

Value

Logical.

Author(s)

Benilton S Carvalho

See Also

require

sampleNames-methods  

Sample names for FeatureSet-like objects

Description

Returns sample names for FeatureSet-like objects.

Methods

object = "FeatureSet"  
Sample names
scqsExample

SnpCnvQSet Example

Description
Example of SnpCnvQSet object.

Usage
data(scqsExample)

Format
Object belongs to SnpCnvQSet class.

Examples
data(scqsExample)
class(scqsExample)

setCluster

DEPRECATED FUNCTIONS. Cluster and large dataset management utilities.

Description
Tools to simplify management of clusters via 'snow' package and large dataset handling through the 'bigmemory' package.

Usage
setCluster(...)getCluster()delCluster()
Details

Some methods in the oligo/crlmm packages, like backgroundCorrect, normalize, summarize and rma can use a cluster (set through 'snow' package). The use of cluster features is conditioned on the availability of the 'bigmemory' (used to provide shared objects across compute nodes) and 'snow' packages.

To use a cluster, 'oligo/crlmm' checks for three requirements: 1) 'ff' is loaded; 2) 'snow' is loaded; and 3) the 'cluster' option is set (e.g., via options(cluster=makeCluster(...)) or setCluster(...)).

If only the 'ff' package is available and loaded (in addition to the caller package - 'oligo' or 'crlmm'), these methods will allow the user to analyze datasets that would not fit in RAM at the expense of performance.

In the situations above (large datasets and cluster), oligo/crlmm uses the options ocSamples and ocProbesets to limit the amount of RAM used by the machine(s). For example, if ocSamples is set to 100, steps like background correction and normalization process (in RAM) 100 samples simultaneously on each compute node. If ocProbesets is set to 10K, then summarization processes 10K probesets at a time on each machine.

Warning

In both scenarios (large dataset and/or cluster use), there is a penalty in performance because data are written to disk (to either minimize memory footprint or share data across compute nodes).

Author(s)

Benilton Carvalho

sfsExample  SnpFeatureSet Example

Description

Example of SnpFeatureSet object.

Usage

data(sfsExample)

Format

Object belongs to SnpFeatureSet class

Examples

data(sfsExample)
class(sfsExample)
**Description**

Utility functions for accessing data in SnpSet objects.

**Usage**

```r
calls(object)
calls(object) <- value
confs(object, transform=TRUE)
confs(object) <- value
```

**Arguments**

- `object`: A SnpSet object.
- `transform`: Logical. Whether to transform the integer representation of the confidence score (for memory efficiency) to a probability. See details.
- `value`: A matrix.

**Details**

`calls` returns the genotype calls. CRLMM stores genotype calls as integers (1 - AA; 2 - AB; 3 - BB).

`confs` returns the confidences associated with the genotype calls. The current implementation of CRLMM stores the confidences as integers to save memory on disk by using the transformation:

\[
\text{round}(-1000\times\log_2(1-p)),
\]

where 'p' is the posterior probability of the call. `confs` is a convenience function that transforms the integer representation back to a probability. Note that if the assayData elements of the SnpSet objects are `ff_matrix` or `ffdf`, the `confs` function will return a warning. For such objects, one should first subset the `ff` object and coerce to a matrix, then apply the above conversion. The function `snpCallProbability` for the `callProbability` slot of SnpSet objects. See the examples below.

`checkOrder` checks whether the object is ordered by chromosome and physical position, evaluating to TRUE or FALSE.

**Note**

Note that the replacement method for `confs<-` expects a matrix of probabilities and will automatically convert the probabilities to an integer representation. See details for the conversion.

The accessor `snpCallProbability` is an accessor for the 'callProbability' element of the `assayData`. The name can be misleading, however, as the accessor will not return a probability if the call probabilities are represented as integers.
See Also

The helper functions \(p2i\) converts probabilities to integers and \(i2p\) converts integers to probabilities. See \code{order} and \code{checkOrder}.

Examples

```r
theCalls <- matrix(sample(1:3, 20, rep=TRUE), nc=2)
p <- matrix(runif(20), nc=2)
integerRepresentation <- matrix(as.integer(round(-1000*log(1-p))), 10, 2)
obj <- new("SnpSet2", call=theCalls, callProbability=integerRepresentation)
calls(obj)
  confs(obj)  ## coerces to probability scale
  int <- Biobase::snpCallProbability(obj)  ## not necessarily a probability
  p3 <- i2p(int)  ## to convert back to a probability
```

---

**SnpSet2-class**  
*Class "SnpSet2"*

**Description**

A container for genotype calls and confidence scores. Similar to the SnpSet class in \pkg{Biobase}, but SnpSet2 extends \code{gset} directly whereas SnpSet extends \code{eset}. Useful properties of \code{gset} include the genome slot and the GenomeAnnotatedDataFrame.

**Objects from the Class**

Objects can be created by calls of the form `new("SnpSet2", assayData, phenoData, featureData, experimentData, annotation, protocolData, call, callProbability, genome, ...)`.

**Slots**

- \code{genome}: Object of class "character" indicating the UCSC genome build. Supported builds are 'hg18' and 'hg19'.
- \code{assayData}: Object of class "AssayData".
- \code{phenoData}: Object of class "AnnotatedDataFrame".
- \code{featureData}: Object of class "AnnotatedDataFrame".
- \code{experimentData}: Object of class "MIAxE".
- \code{annotation}: Object of class "character" --
- \code{protocolData}: Object of class "AnnotatedDataFrame" --
- \code{__classVersion__}: Object of class "Versions" --

**Extends**

Class "\code{gSet}". directly. Class "\code{eSet}". by class "\code{gSet}", distance 2. Class "\code{VersionedBiobase}". by class "\code{gSet}", distance 3. Class "\code{Versioned}". by class "\code{gSet}", distance 4.
Accessors

The argument object for the following methods is an instance of the SnpSet2 class.

calls(object): calls(object) <- value:
   Gets or sets the genotype calls. value can be a matrix or a ff_matrix.

confs(object): confs(object) <- value:
   Gets or sets the genotype confidence scores. value can be a matrix or a ff_matrix.

snpCall(object): snpCallProbability(object) <- value:
   Gets or sets the genotype confidence scores.

Author(s)

R. Scharpf

See Also

SnpSet

Examples

showClass("SnpSet2")
new("SnpSet2")

SnpSuperSet-class  Class "SnpSuperSet"

Description

A class to store locus-level summaries of the quantile normalized intensities, genotype calls, and genotype confidence scores

Objects from the Class

new("SnpSuperSet", alleleA=alleleA, alleleB=alleleB, call=call, callProbability, ...).

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" ~~
protocolData: Object of class "AnnotatedDataFrame" ~~
__classVersion__: Object of class "Versions" ~
splitIndicesByLength

Description
Tools to distribute objects across nodes or by length.

Usage
splitIndicesByLength(x, lg, balance=FALSE)
splitIndicesByNode(x)

Arguments
x object to be split
lg length
balance logical. Currently ignored

Details
splitIndicesByLength splits x in groups of length lg.
splitIndicesByNode splits x in N groups (where N is the number of compute nodes available).
sqsExample

Value
List.

Author(s)
Benilton S Carvalho

See Also
split

Examples

\[
\begin{align*}
x & \leftarrow 1:100 \\
splitIndicesByLength(x, 8) \\
splitIndicesByLength(x, 8, balance=TRUE) \\
splitIndicesByNode(x)
\end{align*}
\]

sqsExample  SnpQSet Example

Description
Example of SnpQSet instance.

Usage
data(sqsExample)

Format
Belongs to SnpQSet class.

Examples
data(sqsExample)
class(sqsExample)
Methods for SummarizedExperiment objects

Description

Methods for SummarizedExperiment.

Usage

```r
## S4 method for signature 'SummarizedExperiment'
baf(object)
## S4 method for signature 'SummarizedExperiment'
chromosome(object,...)
## S4 method for signature 'SummarizedExperiment'
isSnp(object,...)
## S4 method for signature 'SummarizedExperiment'
lrr(object)
```

Arguments

- `object`: A `SummarizedExperiment` object.
- `...`: ignored

Details

- `baf` and `lrr` are accessors for the B allele frequencies and log R ratio assays (matrices or arrays), respectively.
- `chromosome` returns the `seqnames` of the `rowData`.
- `isSnp` returns a logical vector for each marker in `rowData` indicating whether the marker targets a SNP (nonpolymorphic regions are FALSE).

See Also

- `SummarizedExperiment`
Index

+Topic **IO**
  - listCelfiles, 48
+Topic **attribute**
  - getSequenceLengths, 35
+Topic **classes**
  - AlleleSet-class, 4
  - AssayData-methods, 6
  - BeadStudioSet-class, 10
  - BeadStudioSetList-class, 11
  - CNSet-class, 18
  - CopyNumberSet-class, 19
  - DBPDIInfo-class, 24
  - FeatureSet-class, 25
  - ff_matrix-class, 27
  - ff_or_matrix-class, 28
  - ffdf-class, 26
  - GenomeAnnotatedDataFrame-class, 30
  - gSet-class, 38
  - gSetList-class, 39
  - ListClasses, 49
  - SnpSet2-class, 62
  - SnpSuperSet-class, 63
+Topic **datasets**
  - efsExample, 24
  - locusLevelData, 49
  - oligoSet, 53
  - scqsExample, 59
  - sfsExample, 60
  - sqsExample, 65
+Topic **data**
  - pdPkgFromBioC, 55
  - requireAnnotation, 57
+Topic **list**
  - affyPlatforms, 4
+Topic **manip**
  - AssayDataList, 7
  - assayDataList-methods, 8
  - batchStatistics, 9
  - celfilename, 13
  - checkExists, 14
  - checkOrder, 15
  - chromosome2integer, 17
  - CopyNumberSet-methods, 21
  - createFF, 22
  - featureDataList-methods, 25
  - fileConnections, 28
  - flags, 29
  - genomeBuild, 33
  - geometry, 33
  - getA, 34
  - getBar, 35
  - i2p, 40
  - initializeBigMatrix, 41
  - integerMatrix, 42
  - is_ffmatrix, 43
  - isPackageLoaded, 44
  - kind, 45
  - ldSetOptions, 46
  - library2, 47
  - makeFeatureGRanges, 50
  - ocLapply, 51
  - ocSamples, 52
  - parStatus, 54
  - requireClusterPkgSet, 58
  - setCluster, 59
  - SnpSet-methods, 61
  - splitIndicesByLength, 64
+Topic **methods**
  - assayDataList-methods, 8
  - batch, 8
  - batchStatistics, 9
  - chromosome-methods, 16
  - CopyNumberSet-methods, 21
  - db, 23
  - exprs-methods, 25
  - featureDataList-methods, 25
  - flags, 29
GenomeAnnotatedDataFrameFrom-methods, 32
GRanges-methods, 36
isSnp-methods, 45
length-methods, 47
manufacturer-methods, 51
oligoSnipSet-methods, 54
platform-methods, 56
pmFragmentLength-methods, 56
position-methods, 56
sampleNames-methods, 58
SummarizedExperiment-methods, 66

*Topic misc
  affyPlatforms, 4
generics, 30

*Topic utilities
  list.cefies, 48
  [ , CNSet, ANY-method (CNSet-class), 18
  [ , CNSet-method (CNSet-class), 18
  [ , gSetList, ANY-method (gSetList-class), 39
  [ , gSetList-method (gSetList-class), 39
  [ , BafLrrSetList, ANY, ANY-method
    (BeadStudioSetList-class), 11
  [ , BeadStudioSetList, ANY, ANY-method
    (BeadStudioSetList-class), 11
  [ , gSetList, ANY, ANY, BafLrrSet-method
    (BeadStudioSetList-class), 11
  [ , gSetList, ANY, ANY, BafLrrSet-method
    (gSetList-class), 39
  $, gSetList-method (gSetList-class), 39
  $<-, gSetList-method (gSetList-class), 39

  A (getA), 34
  A, AlleleSet-method (getA), 34
  A, CNSet-method (CNSet-class), 18
  A<-, (getA), 34
  A<-, AlleleSet-method (getA), 34
  A<-, CNSet-method (CNSet-class), 18
  AffyExonPDInfo-class (DBPInfo-class), 24
  AffyExonPDInfo-class (DBPInfo-class), 24
  AffyExpressionPDInfo-class (DBPInfo-class), 24
  AffyExpressionPDInfo-class (DBPInfo-class), 24
  AffyGenePDInfo-class (DBPInfo-class), 24
  affyPlatforms, 4
  AffySNPCNVPDInfo-class (DBPInfo-class), 24
  AffySNPCNVPDInfo-class (DBPInfo-class), 24
  AffySNPPPDInfo-class (DBPInfo-class), 24
  AffySTPDInfo-class (DBPInfo-class), 24
  AffySTPDInfo-class (DBPInfo-class), 24
  AffyTilingPDInfo-class (DBPInfo-class), 24
  AffyTilingPDInfo-class (DBPInfo-class), 24
  allele (AlleleSet-class), 4
  allele, AlleleSet-method
    (AlleleSet-class), 4
  allele, CNSet-method (CNSet-class), 18
  allele, SnpFeatureSet-method
    (AlleleSet-class), 4
  AlleleSet, 64
  AlleleSet (AlleleSet-class), 4
  AlleleSet-class, 4
  AnnotatedDataFrame, 30, 31
  annotatedDataFrameFrom, ff_matrix-method
    (ff_matrix-class), 27
  annotation, DBPInfo-method
    (DBPInfo-class), 24
  annotation, gSetList-method
    (gSetList-class), 39
  annotationPacakges, 5
  AssayData, 32
  AssayData-methods, 6
  AssayDataList, 7
  assayDataList (assayDataList-methods), 8
  assayDataList, gSetList-method
    (gSetList-class), 39
  assayDataList, oligoSetList-method
    (assayDataList-methods), 8
  assayDataList-methods, 8
  assayDataNew, 7
  B (getB), 34
  B, AlleleSet-method (getB), 34
  B, CNSet-method (CNSet-class), 18
  B<-, (getB), 34
  B<-, AlleleSet-method (getB), 34
  B<-, CNSet-method (CNSet-class), 18
  baf (generics), 30
baf, BaflrrSetList-method  
   (BeadStudioSetList-class), 11
baf, BeadStudioSet-method  
   (BeadStudioSet-class), 10
baf, BeadStudioSetList-method  
   (BeadStudioSetList-class), 11
baf, oligoSetList-method  
   (BeadStudioSetList-class), 11
baf, oligoSnpsSet-method  
   (oligoSnpsSet-methods), 54
baf, SummarizedExperiment-method  
   (SummarizedExperiment-methods), 66
baf<- (BeadStudioSet-class), 10
baf-, BeadStudioSet-method  
   (BeadStudioSet-class), 10
baf-, oligoSnpsSet-method  
   (oligoSnpsSet-methods), 54
BaflrrSet-class (BeadStudioSet-class), 10
BaflrrSetList-class  
   (BeadStudioSetList-class), 11
batch, 8, 9
batch, CNSet-method (CNSet-class), 18
batchNames, 9
batchNames (batch), 8
batchNames, AssayData-method  
   (AssayData-methods), 6
batchNames, CNSet-method (CNSet-class), 18
batchNames<- (batch), 8
batchNames-, AssayData-method  
   (AssayData-methods), 6
batchNames-, CNSet-method  
   (CNSet-class), 18
batchStatistics, 9, 29
batchStatistics, CNSet-method  
   (CNSet-class), 18
batchStatistics<- (batchStatistics), 9
batchStatistics-, CNSet, AssayData-method  
   (CNSet-class), 18
BeadStudioSet (BeadStudioSet-class), 10
BeadStudioSet-class, 10
BeadStudioSetList, 40
BeadStudioSetList-class, 11
bothStrands (AlleleSet-class), 4
bothStrands, AlleleSet-method  
   (AlleleSet-class), 4
bothStrands, SnpFeatureSet-method  
   (AlleleSet-class), 4
calls (SnpsSet-methods), 61
calls, CNSet-method (CNSet-class), 18
calls, oligoList-method  
   (BeadStudioSetList-class), 11
calls, oligoSnpsSet-method  
   (oligoSnpsSet-methods), 54
calls, SnpsSet-method (SnpsSet-methods), 61
calls, SnpsSet2-method (SnpsSet2-class), 62
calls<- (SnpsSet-methods), 61
calls<-, CNSet, matrix-method  
   (CNSet-class), 18
calls<-, oligoSnpsSet, matrix-method  
   (oligoSnpsSet-methods), 54
calls<-, SnpsSet, matrix-method  
   (SnpsSet-methods), 61
calls<-, SnpsSet2, matrix-method  
   (SnpsSet2-class), 62
callsConfidence, oligoSnpsSet-method  
   (oligoSnpsSet-methods), 54
callsConfidence<-, oligoSnpsSet, matrix-method  
   (oligoSnpsSet-methods), 54
celfileDate, 12
celfilName, 13
checkExists, 14
checkOrder, 15, 62
checkOrder, CopyNumberSet-method  
   (CopyNumberSet-class), 19
checkOrder, gSet-method (gSet-class), 38
checkOrder, SnpsSet-method  
   (SnpsSet-methods), 61
chromosome, 39
chromosome (chromosome-methods), 16
chromosome, AnnotatedDataFrame-method  
   (chromosome-methods), 16
chromosome, GenomeAnnotatedDataFrame-method  
   (chromosome-methods), 16
chromosome, GRanges-method  
   (chromosome-methods), 16
chromosome, GRangesList-method  
   (chromosome-methods), 16
chromosome, gSet-method  
   (chromosome-methods), 16
chromosome, gSetList-method  
   (gSetList-class), 39
chromosome, SnpsSet-method  
   (chromosome-methods), 16
coerce, AnnotatedDataFrame, GenomeAnnotatedDataFrame (GenomeAnnotatedDataFrame-methods), 6
coerce, BeadStudioSet, data.frame-method (BeadStudioSet-class), 10
coerce, CNSet, CopyNumberSet-method (CNSet-class), 18
coerce, CNSet, oligoSnSet (oligoSnSet-methods), 18
coerce, CNSet, oligoSnSet-method (oligoSnSet-class), 54
corr, CNSet, character-method (CNSet-class), 18
coverage2 (GRanges-methods), 36
coverage2, GRanges-method (GRanges-methods), 36
coverage2, GRangesList-method

chromosome, SummarizedExperiment-method (SummarizedExperiment-methods), 66
chromosome-methods, 16
chromosome2, integer, 16, 17
chromosome<-(chromosome-methods), 16
chromosome<-, GenomeAnnotatedDataFrame, integer-method (chromosome-methods), 16
chromosome<-, gSet, integer-method (chromosome-methods), 16
chromosome<-, SnpSet, integer-method (chromosome-methods), 16
chromosomePositionOrder (checkOrder), 15
close (fileConnections), 28
close, AlleleSet-method (getA), 34
close, array-method (fileConnections), 28
close, CNSet-method (CNSet-class), 18
close, matrix-method (fileConnections), 28
close, numeric-method (fileConnections), 28
closeff (fileConnections), 28
closeff, CNSet-method (fileConnections), 28
cnConfidence (CopyNumberSet-methods), 21
cnConfidence, CopyNumberSet-method (CopyNumberSet-class), 19
cnConfidence, oligoSnSet-method (oligoSnSet-methods), 54
cnConfidence<-(CopyNumberSet-methods), 21
cnConfidence<-, CopyNumberSet, matrix-method (CopyNumberSet-class), 19
cnConfidence<-, oligoSnSet, matrix-method (oligoSnSet-methods), 54
CNSet, 5
CNSet (CNSet-class), 18
CNSet-class, 18
closeff, fileConnections (getA), 34
coverage2 (GRanges-methods), 36
coverage2, GRanges-method (GRanges-methods), 36
coverage2, GRangesList-method
(GRanges-methods), 36
createFF, 22

db, 23
db, AlleleSet-method (AlleleSet-class), 4
db, DBPDInfo-method (db), 23
db, FeatureSet-method (db), 23
db, gSet-method (gSet-class), 38
db, SnpCnvQSet-method (db), 23
db, SnpQSet-method (db), 23
db, SnpSet-method (db), 23
DBPDInfo (DBPDInfo-class), 24
DBPDInfo-class, 24
delCluster (setCluster), 59
delCluster-deprecated (setCluster), 59
dims, gSetList-method (gSetList-class), 39

efsExample, 24
elementLengths, gSetList-method
 (gSetList-class), 39
eSet, 5, 10, 20, 26, 38, 62, 64
ExonFeatureSet (FeatureSet-class), 25
ExonFeatureSet-class
 (FeatureSet-class), 25
ExpressionFeatureSet
 (FeatureSet-class), 25
ExpressionFeatureSet-class
 (FeatureSet-class), 25
ExpressionPDInfo (DBPDInfo-class), 24
ExpressionPDInfo-class
 (DBPDInfo-class), 24
exprs, FeatureSet-method
 (exprs-methods), 25
exprs, SnpSet-method (SnpSet2-class), 62
exprs-methods, 25

featureDataList
 (featureDataList-methods), 25
featureDataList, gSetList-method
 (gSetList-class), 39
featureDataList-methods, 25
FeatureSet (FeatureSet-class), 25
FeatureSet-class, 25
ff, 28
ff, matrix-class, 27
ff, or_matrix-class, 28
ffdf, 28

ffdf-class, 26
fileConnections, 28
findOverlaps, 50
findOverlaps, GRanges, gSet-method
 (GRanges-methods), 36
findOverlaps, GRangesList, gSet-method
 (GRanges-methods), 36
flags, 29
flags, AssayData-method
 (AssayData-methods), 6
flags, CNSet-method (CNSet-class), 18

GeneFeatureSet (FeatureSet-class), 25
GeneFeatureSet-class
 (FeatureSet-class), 25
generics, 30
GenomeAnnotatedDataFrame, 32, 50
GenomeAnnotatedDataFrame
 (GenomeAnnotatedDataFrame-class), 30
GenomeAnnotatedDataFrame-class, 30
GenomeAnnotatedDataFrameFrom
 (GenomeAnnotatedDataFrameFrom-methods), 32
GenomeAnnotatedDataFrameFrom, array-method
 (GenomeAnnotatedDataFrameFrom-methods), 32
GenomeAnnotatedDataFrameFrom, AssayData-method
 (GenomeAnnotatedDataFrameFrom-methods), 32
GenomeAnnotatedDataFrameFrom, ff_or_matrix-method
 (GenomeAnnotatedDataFrameFrom-methods), 32
GenomeAnnotatedDataFrameFrom, list-method
 (GenomeAnnotatedDataFrameFrom-methods), 32
GenomeAnnotatedDataFrameFrom, NULL-method
 (GenomeAnnotatedDataFrameFrom-methods), 32
GenomeAnnotatedDataFrameFrom-methods, 32
genomeBuild, 33
genomeBuild, DBPDInfo-method
 (genomeBuild), 33
genomeBuild, FeatureSet-method
 (genomeBuild), 33
genomeBuild, GRanges-method
 (GRanges-methods), 36
genomeBuild, gSet-method (gSet-class), 38
genomeBuild, gSetList-method
   (gSetList-class), 39
initialize, CopyNumberSet-method
   (CopyNumberSet-class), 19
initialize, DBPDIInfo-method
   (DBPDIInfo-class), 19
initialize, eSetList-method
   (ListClasses), 49
initialize, FeatureSet-method
   (FeatureSet-class), 25
initialize, GenomeAnnotatedDataFrame-method
   (GenomeAnnotatedDataFrame-class), 30
initialize, gSet-method (gSet-class), 38
initialize, gSetList-method
   (gSetList-class), 38
initialize, gSet-method (gSet-class), 39
initialize, gSetSuperSet-method
   (gSetSuperSet-class), 63
initializeBigArray
   (initializeBigMatrix), 41
initializeBigMatrix, 41
initializeBigVector
   (initializeBigMatrix), 41
integer2chromosome
   (chromosome2integer), 17
integerArray (integerMatrix), 42
integerMatrix, 42
is.ffmatrix, 43
isPackageLoaded, 44
is.glm, 39
is.glm, (is.glm-methods), 45
is.glm, character-method (is.glm-methods), 45
is.glm, GenomeAnnotatedDataFrame-method
   (is.glm-methods), 45
is.glm, gSet-method (is.glm-methods), 45
is.glm, gSet-method (is.glm-methods), 45
is.glm, SnpsSuperSet-method
   (is.glm-methods), 63
is.glm, SummarizedExperiment-method
   (SummarizedExperiment-methods), 66
is.glm-methods, 45
kind, 45
kind, AffyExonPDInfo-method (kind), 45
kind, AffyExpressionPDInfo-method
   (kind), 45
kind, AffyGenePDInfo-method (kind), 45

getA, 34
getA, AlleleSet-method
   (AlleleSet-class), 4
getA, SnpCnvQSet-method (getA), 34
getA, SnpQSet-method (getA), 34
getA, TilingFeatureSet2-method (getA), 34
getArm (gSet-class), 38
getArm, GenomeAnnotatedDataFrame-method
   (GenomeAnnotatedDataFrame-class), 30
getArm, gSet-method (gSet-class), 38
getBar, 35
getCluster (setCluster), 59
getCluster-deprecated (setCluster), 59
getM (getA), 34
getM, AlleleSet-method
   (AlleleSet-class), 4
getM, SnpCnvQSet-method (getA), 34
getM, SnpQSet-method (getA), 34
getM, TilingFeatureSet2-method (getA), 34
getSequenceLengths, 35
GRanges, 37, 50
GRanges-methods, 36
gSet, 10, 62
gSet (gSet-class), 38
gSet-class, 38
gSetList-class, 39

i2p, 40, 62
initialize, BeadStudioSet-method
   (BeadStudioSet-class), 10
initialize, BeadStudioSetList-method
   (BeadStudioSetList-class), 11
initialize, CNSSet-method (CNSSet-class), 18
initialize, CNSSetLM-method
   (CNSSet-class), 18

geometry, DBPDIInfo-method (geometry), 33
getArm (gSet-class), 38
getArm, gSet-method (gSet-class), 38
getBar, 35
getCluster (setCluster), 59
getCluster-deprecated (setCluster), 59
getM (getA), 34
getM, AlleleSet-method
   (AlleleSet-class), 4
getM, SnpCnvQSet-method (getA), 34
getM, SnpQSet-method (getA), 34
getM, TilingFeatureSet2-method (getA), 34
getSequenceLengths, 35
GRanges, 37, 50
GRanges-methods, 36
gSet, 10, 62
gSet (gSet-class), 38
gSet-class, 38
gSetList-class, 39

i2p, 40, 62
initialize, BeadStudioSet-method
   (BeadStudioSet-class), 10
initialize, BeadStudioSetList-method
   (BeadStudioSetList-class), 11
initialize, CNSSet-method (CNSSet-class), 18
initialize, CNSSetLM-method
   (CNSSet-class), 18
initialize, CopyNumberSet-method
   (CopyNumberSet-class), 19
initialize, DBPDIInfo-method
   (DBPDIInfo-class), 19
initialize, eSetList-method
   (ListClasses), 49
initialize, FeatureSet-method
   (FeatureSet-class), 25
initialize, GenomeAnnotatedDataFrame-method
   (GenomeAnnotatedDataFrame-class), 30
initialize, gSet-method (gSet-class), 38
initialize, gSetList-method
   (gSetList-class), 38
initialize, oligoSnpSet-method
   (oligoSnpSet-methods), 54
initialize, SnpSet2-method
   (SnpSet2-class), 62
initialize, SnpsSuperSet-method
   (SnpsSuperSet-class), 63
initializeBigArray
   (initializeBigMatrix), 41
initializeBigMatrix, 41
initializeBigVector
   (initializeBigMatrix), 41
integer2chromosome
   (chromosome2integer), 17
integerArray (integerMatrix), 42
integerMatrix, 42
is.ffmatrix, 43
isPackageLoaded, 44
is.glm, 39
is.glm, (is.glm-methods), 45
is.glm, character-method (is.glm-methods), 45
is.glm, GenomeAnnotatedDataFrame-method
   (is.glm-methods), 45
is.glm, gSet-method (is.glm-methods), 45
is.glm, gSet-method (is.glm-methods), 45
is.glm, SnpsSuperSet-method
   (is.glm-methods), 63
is.glm, SummarizedExperiment-method
   (SummarizedExperiment-methods), 66
is.glm-methods, 45
kind, 45
kind, AffyExonPDInfo-method (kind), 45
kind, AffyExpressionPDInfo-method
   (kind), 45
kind, AffyGenePDInfo-method (kind), 45
INDEX

kind, AffyHTAPDInfo-method (kind), 45
kind, AffySNPCNVPDInfo-method (kind), 45
kkind, AffySNPDPDInfo-method (kind), 45
kind, ExpressionPDInfo-method (kind), 45
kind, FeatureSet-method (kind), 45
kind, TilingPDInfo-method (kind), 45

ldPath (ldSetOptions), 46
ldSetOptions, 46
ldStatus (ldSetOptions), 46
length, FeatureSet-method
(length-methods), 47
length, gSetList-method
gSetList-class, 39
length-methods, 47
library, 47
library2, 47
list.cufiles, 48
list.files, 48
list_or_ffdf, 27
list_or_ffdf-class (ffdf-class), 26
ListClasses, 49
locusLevelData, 49, 53
lrr (generics), 30
lrr, BafLrrSetList-method
(BeadStudioSetList-class), 11
lrr, BeadStudioSet-method
(BeadStudioSet-class), 10
lrr, BeadStudioSetList-method
(BeadStudioSetList-class), 11
lrr, SummarizedExperiment-method
(SummarizedExperiment-methods), 66
lrr <<- (BeadStudioSet-class), 10
lrr <<-, BafLrrSet, ANY-method
(BeadStudioSet-class), 10
lrr <<-, BafLrrSet-method
(BeadStudioSet-class), 10
lrr <<-, BafLrrSetList, matrix-method
(BeadStudioSetList-class), 11
lrr <<-, BeadStudioSet, ANY-method
(BeadStudioSet-class), 10
lrr <<-, BeadStudioSet-method
(BeadStudioSet-class), 10

makeFeatureGRanges, gSet-method
(gSet-class), 38
makeFeatureGRanges, gSetList-method
gSetList-class, 39
manufacturer (manufacturer-methods), 51
manufacturer, DBPDInfo-method
(manufacturer-methods), 51
manufacturer, FeatureSet-method
(manufacturer-methods), 51
matrix, 32

NgsExpressionPDInfo (DBPDInfo-class), 24
NgsExpressionPDInfo-class
(DBPDInfo-class), 24
NgsTilingPDInfo (DBPDInfo-class), 24
NgsTilingPDInfo-class (DBPDInfo-class), 24

nu (AssayData-methods), 6
nu, AssayData, character-method
(AssayData-methods), 6
nu, CNSet, character-method
(CNSet-class), 18
numberProbes (GRanges-methods), 36
numberProbes, GRanges-method
(GRanges-methods), 36
numberProbes, GRangesList-method
(GRanges-methods), 36

oclapply, 51
ocProbesets (ocSamples), 52
ocSamples, 52
oldClass, 27
oligoSet, 53
oligoSetList, 40
oligoSetList-class
(BeadStudioSetList-class), 11
oligoSnpsSet, 20
oligoSnpsSet-class
(oligoSnpsSet-methods), 54
oligoSnpsSet-methods, 54
open (fileConnections), 28
open, AlleleSet-method (getA), 34
open, array-method (fileConnections), 28
open, CNSet-method (CNSet-class), 18
open, matrix-method (fileConnections), 28
openff (fileConnections), 28
openffCNSet-method (fileConnections), 28
order, J5, 62
p2i, 62
p2i (12p), 40
parStatus, 54
pdfPkgFromBioC, 55
phi (AssayData-methods), 6
phi, AssayData, character-method
(AssayData-methods), 6
phi, CNSet, character-method
(CNSet-class), 18
platform (platform-methods), 56
platform, FeatureSet-method
(platform-methods), 56
platform-methods, 56
pmFragmentLength
(pmgFragmentLength-methods), 56
pmFragmentLength, AffySNPPDInfo-method
(pmgFragmentLength-methods), 56
pmFragmentLength-methods, 56
position, 39
position (position-methods), 56
position, AnnotatedDataFrame-method
(position-methods), 56
position, GenomeAnnotatedDataFrame-method
(position-methods), 56
position, gSet-method
(position-methods), 56
position, gSetList-method
(gSetList-class), 39
position, SnpSet-method
(position-methods), 56
position-methods, 56
position<-
(GenomeAnnotatedDataFrame-class), 30
position<-, GenomeAnnotatedDataFrame, integer-method
(GenomeAnnotatedDataFrame-class), 30
position<-, oligoSnpSet, integer-method
(oligoSnpSet-methods), 54
read.celfiles, 25
read.xysfiles, 25
requireAnnotation, 57
requireClusterPkg
(requireClusterPkgSet), 58
requireClusterPkg-deprecated
(requireClusterPkgSet), 58
requireClusterPkgSet, 58
requireClusterPkgSet-deprecated
(requireClusterPkgSet), 58
sampleNames, FeatureSet-method
(sampleNames-methods), 58
sampleNames, GRanges-method
(GRanges-methods), 36
sampleNames, GRangesList-method
(GRanges-methods), 36
sampleNames, gSetList-method
(gSetList-class), 39
sampleNames-methods, 58
sampleNames<-, gSetList, character-method
(gSetList-class), 39
sqsExample, 59
se.exprs, FeatureSet-method
(exprs-methods), 25
setCluster, 59
setCluster-deprecated (setCluster), 59
sfsExample, 60
show, BeadStudioSet-method
(BeadStudioSet-class), 10
show, CNSet-method (CNSet-class), 18
show, DBPInfo-method (DBPInfo-class), 24
show, FeatureSet-method
(FeatureSet-class), 25
show, gSet-method (gSet-class), 38
show, gSetList-method (gSetList-class), 39
sigma2, CNSet, character-method
(CNSet-class), 18
snpCallProbability, 61
snpCallProbability, CNSet-method
(CNSet-class), 18
SnpCallFeatureSet (FeatureSet-class), 25
SnpCnvFeatureSet-class
(FeatureSet-class), 25
SNPCNVPDInfo (DBPInfo-class), 24
SNPCNVPDInfo-class (DBPInfo-class), 24
SnpFeatureSet (FeatureSet-class), 25
SnpFeatureSet-class (FeatureSet-class), 25
SNPPDInfo (DBPInfo-class), 24
SNPPDInfo-class (DBPInfo-class), 24
snpRma, 35
INDEX

SnpSet, 61, 63, 64
SnpSet-methods, 61
SnpSet-class, 62
SnpSuperSet, 5
SnpSuperSet (SnpSuperSet-class), 63
SnpSuperSet-class, 63
splitIndicesByLength, 64
splitIndicesByNode
(splitIndicesByLength), 64
sqsExample, 65
state (GRanges-methods), 36
state, GRanges-method (GRanges-methods), 36
state, GRangesList-method
(GRanges-methods), 36
SummarizedExperiment, 66
SummarizedExperiment-methods, 66
tau2,CNSet.character-method
(CNSet-class), 18
TilingFeatureSet (FeatureSet-class), 25
TilingFeatureSet-class
(FeatureSet-class), 25
TilingFeatureSet2 (FeatureSet-class), 25
TilingFeatureSet2-class
(FeatureSet-class), 25
TilingPDInfo (DBPDPDInfo-class), 24
TilingPDInfo-class (DBPDPDInfo-class), 24
updateObject.BeadStudioSet-method
(BeadStudioSet-class), 10
updateObject.BeadStudioSetList-method
(BeadStudioSetList-class), 11
updateObject.CNSet-method
(CNSet-class), 18
updateObject.GenomeAnnotatedDataFrame-method
(GenomeAnnotatedDataFrame-class), 30
updateObject.oligoSnpSet-method
(oligoSnpSet-methods), 54

Versioned, 5, 10, 20, 26, 30, 38, 62, 64
VersionedBlobase, 5, 10, 20, 26, 38, 62, 64