# Package 'oligoClasses'

December 2, 2025

```
Version 1.72.0
Title Classes for high-throughput arrays supported by oligo and crlmm
Author Benilton Carvalho and Robert Scharpf
Maintainer Benilton Car-
     valho <beniltoncarvalho@gmail.com> and Robert Scharpf <rscharpf@jhsph.edu>
Depends R (>= 2.14)
Imports BiocGenerics (>= 0.27.1), Biobase (>= 2.17.8), methods,
     graphics, IRanges (>= 2.5.17), GenomicRanges (>= 1.23.7),
     SummarizedExperiment, Biostrings (>= 2.23.6), affyio (>=
     1.23.2), foreach, BiocManager, utils, S4Vectors (>= 0.9.25),
     RSOLite, DBI, ff
Enhances doMC, doMPI, doSNOW, doParallel, doRedis
Suggests hapmapsnp5, hapmapsnp6, pd.genomewidesnp.6,
     pd.genomewidesnp.5, pd.mapping50k.hind240,
     pd.mapping50k.xba240, pd.mapping250k.sty, pd.mapping250k.nsp,
     genomewidesnp6Crlmm (>= 1.0.7), genomewidesnp5Crlmm (>= 1.0.6),
     RUnit, human370v1cCrlmm, VanillaICE, crlmm
Description This package contains class definitions, validity checks, and initialization meth-
     ods for classes used by the oligo and crlmm packages.
License GPL (>= 2)
LazyLoad yes
Collate AllClasses.R AllGenerics.R utils-general.R utils-lds.R
     utils-parallel.R methods-gSet.R initialize-methods.R
     methods-AlleleSet.R methods-AnnotatedDataFrame.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
## time-stamp-pattern ``8/Date: %3a %3b %2d %02H:%02M:%02S %Z %:y\n"
RoxygenNote 6.1.1
```

git\_url https://git.bioconductor.org/packages/oligoClasses

2 Contents

git_branch RELEASE_3_22
git_last_commit d8ec574
git_last_commit_date 2025-10-29
<b>Repository</b> Bioconductor 3.22
Date/Publication 2025-12-01

## **Contents**

affyPlatforms	3
AlleleSet-class	4
annotationPackages	5
AssayData-methods	5
AssayDataList	6
assayDataList-methods	7
batch	7
batchStatistics	8
BeadStudioSet-class	9
BeadStudioSetList-class	10
celfileDate	11
celfileName	11
checkExists	12
checkOrder	13
chromosome-methods	14
chromosome2integer	15
	15
	17
CopyNumberSet-methods	18
createFF	20
db	20
DBPDInfo-class	21
Deprecated	21
efsExample	22
exprs-methods	22
featureDataList-methods	22
	23
ffdf-class	24
	24
ff_or_matrix-class	25
fileConnections	25
	26
	27
~	27
GenomeAnnotatedDataFrameFrom-methods	28
	29
	30
	30
	31
<u> </u>	32
	33
	34
O	

affyPlatforms 3

ex		61
	SummarizedExperiment-methods	59
	sqsExample	
	splitIndicesByLength	
	SnpSuperSet-class	
	SnpSet2-class	
	SnpSet-methods	
	sfsExample	
	setCluster	
	scqsExample	
	sampleNames-methods	
	requireClusterPkgSet	52
	requireAnnotation	
	position-methods	
	pmFragmentLength-methods	50
	platform-methods	50
	pdPkgFromBioC	49
	parStatus	49
	oligoSnpSet-methods	
	oligoSet	47
	ocSamples	46
	ocLapply	46
	manufacturer-methods	45
	makeFeatureGRanges	
	locusLevelData	44
	ListClasses	
	list.celfiles	
	library2	
	length-methods	
	ldSetOptions	
	kind	
	isSnp-methods	
	isPackageLoaded	
	is.ffmatrix	
	integerMatrix	
	initializeBigMatrix	
	gSetList-class	
	gSetList-class	35

## Description

Provides a listing of available Affymetrix platforms currently supported by the R package oligo

### Usage

affyPlatforms()

4 AlleleSet-class

#### Value

A vector of class character.

#### Author(s)

R. Scharpf

#### **Examples**

```
affyPlatforms()
```

AlleleSet-class

Class "AlleleSet"

### Description

A class for storing the locus-level summaries of the normalized intensities

#### **Objects from the Class**

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

#### **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" ~~
```

#### **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)
```

annotationPackages 5

#### Author(s)

R. Scharpf

#### See Also

```
SnpSuperSet, CNSet
```

### **Examples**

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

AssayData-methods

Methods for class AssayData in the oligoClasses package

#### **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"): ...</pre>
```

6 AssayDataList

#### **Details**

1M: 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**

```
library(crlmm)
library(Biobase)
data(cnSetExample, package="crlmm")
cnSet <- cnSetExample</pre>
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, ]
\#\# 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, ]
```

 ${\tt AssayDataList}$ 

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

assayDataList-methods 7

#### Author(s)

R.Scharpf

#### See Also

assayDataNew

### **Examples**

```
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)</pre>
```

assayDataList-methods 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</pre>
```

#### **Arguments**

object An object of class CNSet.

value For 'batchNames', the value must be a character string corresponding of the

unique batch names.

8 batchStatistics

#### 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**

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

batchStatistics

Accessor for batch statistics uses for copy number estimation and storage of model parameters

### Description

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

### Usage

```
batchStatistics(object)
batchStatistics(object) <- value</pre>
```

### **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

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

### See Also

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

BeadStudioSet-class 9

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 "MIAxE" ~~
annotation: Object of class "character" ~~
protocolData: Object of class "AnnotatedDataFrame" ~~
genome: Object of class "character" ~~
.__classVersion__: 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.

```
{\tt copyNumber(object): accessor \ for \ log \ R \ ratios.}
```

 $copyNumber(object) \leftarrow value: replacement method for the log R ratios$ 

initialize signature(.0bject = "BeadStudioSet"): constructs an instance of the class

1rr(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.</pre>

show(object): print a short summary of the BeadStudioSet object.

updateObject(object): update a BeadStudioSet object.

10 BeadStudioSetList-class

#### Author(s)

R. Scharpf

#### **Examples**

```
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

```
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

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.

1rr(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.

celfileDate 11

#### Author(s)

R. Scharpf

#### See Also

See supporting packages for methods defined for the class.

celfileDate

Cel file dates

### **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(hapmapsnp6)
path <- system.file("celFiles", package="hapmapsnp6")
celfiles <- list.celfiles(path, full.names=TRUE)
dts <- sapply(celfiles, celfileDate)</pre>
```

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

12 checkExists

#### 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**

```
## 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.</name>
.FUN2	Not currently used.
.save.it	Logical. Whether to save the object to the directory indicaged 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.

checkOrder 13

### 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)
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)</pre>
```

checkOrder

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

### **Description**

Checks whether a eSet-derived class (e.g., a SnpSet or CNSet object) is ordered by chromosome and physical position

### Usage

```
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

14 chromosome-methods

#### Author(s)

R. Scharpf

#### See Also

order

### **Examples**

```
data(oligoSetExample)
if(!checkOrder(oligoSet)){
  oligoSet <- chromosomePositionOrder(oligoSet)
}
checkOrder(oligoSet)</pre>
```

chromosome-methods

Methods for function chromosome in package oligoClasses

### **Description**

Methods for function chromosome in package oligoClasses ~~

#### Methods

The methods for chromosome extracts the chromosome (represented as an integer) for each marker in a 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

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

chromosome2integer 15

chromosome2integer

Converts chromosome to integer

### **Description**

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

### Usage

```
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**

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

CNSet-class

Class "CNSet"

### **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.

16 CNSet-class

#### **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.

new(CNSet, alleleA=new("matrix"), alleleB=new("matrix"), call=new("matrix"), callProbability=new('batch=new("factor"))

#### **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(objet): 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(objet): 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</pre>

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, "oligoSnpSet"): coerce a CNSet object to an object of class oligoSnpSet – a container for the total copy number and genotype calls.

corr(object): the correlation of the A and B intensities within each genotype.

CopyNumberSet-class

```
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'.

open(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, annotation, protocolData, copyNumber, cnConfidence, ...).

#### 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(.0bject = "CopyNumberSet"): ...</pre>
```

#### 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**

```
showClass("CopyNumberSet")
cnset <- new("CopyNumberSet")
ls(Biobase::assayData(cnset))</pre>
```

CopyNumberSet-methods Methods for class CopyNumberSet.

#### **Description**

Accessors and CopyNumberSet

### Usage

```
copyNumber(object, ...)
cnConfidence(object)
copyNumber(object) <- value
cnConfidence(object) <- value</pre>
```

### **Arguments**

```
object CopyNumberSet object or derived class
... Ignored for CopyNumberSet and oligoSnpSet.
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.

```
library(Biobase)
data(locusLevelData)
path <- system.file("extdata", package="oligoClasses")</pre>
fd <- readRDS(file.path(path, "genomeAnnotatedDataFrameExample.rds"))</pre>
## the following command creates an 'oligoSnpSet' 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("oligoSnpSet",</pre>
  copyNumber=integerMatrix(log2(locusLevelData[["copynumber"]]/100), 100),
  call=locusLevelData[["genotypes"]],
  callProbability=integerMatrix(locusLevelData[["crlmmConfidence"]], 1),
  annotation=locusLevelData[["platform"]],
 featureData=fd,
 genome="hg19")
## There are several accessors for the oligoSnpSet class.
icn <- copyNumber(oligoSet)</pre>
range(icn) ## integer scale
lcn <- icn/100
range(lcn) ## log2 copy number
## confidence scores for the genotypes are also represented on an integer scale
ipr <- snpCallProbability(oligoSet)</pre>
range(ipr) ## integer scale
## for genotype confidence scores, the helper function i2p
## converts back to a probability scale
pr <- i2p(ipr)</pre>
range(pr)
## The helper function confs is a shortcut, extracting the
## integer-based confidence scores and transforming to the
## probability scale
pr2 <- confs(oligoSet)</pre>
all.equal(pr, pr2)
## To extract information on the annotation of the SNPs, one can use
position(oligoSet)
chromosome(oligoSet)
## the position and chromosome coordinates were extracted from build hg19
genomeBuild(oligoSet)
```

20 db

createFF Create ff objects.

### Description

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

### Usage

```
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.

DBPDInfo-class 21

#### 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)
```

DBPDInfo-class

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

Deprecated

oligoClasses Deprecated

### **Description**

The function, class, or data object you asked for has been deprecated.

22 featureDataList-methods

efsExample

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 23

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, ...). But 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(.0bject = "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

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

24 ff\_matrix-class

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.

ff\_matrix-class

Class "ff\_matrix"

### Description

```
\sim A concise (1-5 lines) description of what the class is. \sim
```

### **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"): ...
```

```
showClass("ff_matrix")
```

ff\_or\_matrix-class 25

```
ff_or_matrix-class Class "ff_or_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**

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

file Connections

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

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

### Arguments

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

#### Value

not applicable

26 flags

### 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

generics 27

generics

Miscellaneous generics. Methods defined in packages that depend on oligoClasses

### **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

```
GenomeAnnotatedDataFrame-class
```

Class "GenomeAnnotatedDataFrame"

### **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.

#### 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</pre>
```

Get or set chromosome.

```
isSnp(object):
```

Many platforms include polymorphic and nonpolymorphic markers. isSnp evalutes to TRUE if the marker is polymorphic.

```
position(ojbect):
```

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

 ${\tt GenomeAnnotatedDataFrameFrom-methods}$ 

 $\label{lem:methods} \textit{Methods for Function GenomeAnnotatedDataFrameFrom in Package oligoClasses}$ 

### **Description**

 ${\tt GenomeAnnotatedDataFrameFrom\ is\ a\ convenience\ for\ creating\ {\tt GenomeAnnotatedDataFrame\ objects}.}$ 

#### 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

genomeBuild 29

### **Examples**

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

```
genomeBuild(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'.

```
showMethods("genomeBuild", where="package:oligoClasses")
```

30 getA

geometry

Array Geometry Information

#### **Description**

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

#### Usage

```
geometry(object)
```

#### **Arguments**

object

PDInfo or FeatureSet object

### **Examples**

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

getA

Compute average log-intensities / log-ratios

#### **Description**

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

### Usage

```
getA(object)
getM(object)
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.)
- senseThataBsense allele B. (Not applicable for Affymetrix 5.0 and 6.0 platforms.)
- alleleAAffymetrix 5.0 and 6.0 platforms

getBar 31

• alleleBAffymetrix 5.0 and 6.0 platforms

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

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.

#### 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

```
message(getBar())
```

32 getSequenceLengths

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

```
getSequenceLengths("hg18")
getSequenceLengths("hg19")
if(require("GenomicRanges")){
## from GenomicRanges
 sl <- getSequenceLengths("hg18")[c("chr1", "chr2", "chr3")]</pre>
 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)),
   Rle(strand(c("-", "+", "*", "+", "-")).
       c(1, 2, 2, 3, 2)),
   score = 1:10,
   GC = seq(1, 0, length=10),
   seqlengths=sl)
 metadata(gr) <- list(genome="hg18")</pre>
metadata(gr)
}
```

GRanges-methods 33

GRanges-methods

Methods for GRanges objects

#### **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** 

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

gr2 <- GRanges(seqnames = c("chr1", "chr1"),
    ranges = IRanges(c(7,13), width = 3),
    state=c(2L, 2L), numberProbes=c(200L, 250L))
gr3 <- GRanges(seqnames = c("chr1", "chr2"),
    ranges = IRanges(c(1, 4), c(3, 9)),</pre>
```

34 gSet-class

```
state=c(1L, 4L), numberProbes=c(300L, 350L))
## Ranges organized by sample
grl <- GRangesList("sample1" = gr1, "sample2" = gr2, "sample3" = gr3)
sampleNames(grl) ## same as names(grl)
numberProbes(grl)
chromosome(grl)
state(grl)
gr <- stack(grl)
sampleNames(gr)
chromosome(gr)
state(gr)</pre>
```

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.
```

```
db(object): database connection
```

gSetList-class 35

```
genomeBuild(object), genomeBuild(object) <- value:</pre>
         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" ~~
```

i2p

#### 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

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

```
genomeBuild(object), genomeBuild(object) <- value:</pre>
```

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")
```

i2p

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

### **Description**

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.

### Usage

```
i2p(i)
```

p2i(p)

initializeBigMatrix 37

# 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 2pi is as.integer(-1000*log(1-p))
```

#### See Also

confs

# **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

38 integerMatrix

#### **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**

```
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))</pre>
```

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

```
integerMatrix(x, scale = 100)
integerArray(x, scale=100)
```

## 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)</pre>
```

is.ffmatrix 39

is.ffmatrix

Check if object is an ff-matrix object.

# **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.

# **Examples**

```
if (isPackageLoaded("ff")){
   x1 <- ff(vmode="double", dim=c(10, 2))
   is.ffmatrix(x1)
}
x1 <- matrix(0, nr=10, nc=2)
is.ffmatrix(x1)</pre>
```

 $is {\tt PackageLoaded}$ 

Check if package is loaded.

# 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.

40 kind

#### Value

Logical.

## See Also

search

# **Examples**

```
isPackageLoaded("oligoClasses")
isPackageLoaded("ff")
isPackageLoaded("snow")
```

isSnp-methods

Methods for Function isSnp in package oligoClasses~~

## **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"genature(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

IdSetOptions 41

#### Value

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

## **Examples**

```
if (require(pd.mapping50k.xba240)){
  data(sfsExample)
  Biobase::annotation(sfsExample) <- "pd.mapping50k.xba240"
  kind(sfsExample)
}</pre>
```

ldSetOptions

Set/check large dataset options.

#### **Description**

Set/check large dataset options.

## Usage

```
ldSetOptions(nsamples=100, nprobesets=20000, path=getwd(), verbose=FALSE)
ldStatus(verbose=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.

1dStatus checks the support for large datasets.

1dPath checks where ff files are stored.

## Author(s)

Benilton S Carvalho

#### See Also

ocSamples, ocProbesets

library2

# **Examples**

ldStatus(TRUE)

length-methods

Number of samples for FeatureSet-like objects.

# Description

Number of samples for FeatureSet-like objects.

# Methods

**x** = "FeatureSet" Number of samples

library2

Supress package startup messages when loading a library

# Description

Supress 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 43

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())
}</pre>
```

ListClasses

eSetList class

# **Description**

Initialization method for eSetList virtual class.

makeFeatureGRanges

locusLevelData

Basic data elements required for the HMM

## **Description**

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

#### **Usage**

```
data(locusLevelData)
```

#### **Format**

A list

#### **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 locusLevelData is a list that contains (1), (2), (4), and (5).

# 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.

# **Examples**

```
data(locusLevelData)
str(locusLevelData)
```

makeFeatureGRanges

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, ...)
```

manufacturer-methods 45

## **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

 $\verb|findOverlaps|, GRanges|, Genome Annotated Data Frame|$ 

## **Examples**

```
library(oligoClasses)
library(GenomicRanges)
library(Biobase)
library(foreach)
registerDoSEQ()
data(oligoSetExample, package="oligoClasses")
oligoSet <- oligoSet[chromosome(oligoSet) == 1, ]
makeFeatureGRanges(oligoSet)</pre>
```

manufacturer-methods Manufacturer ID for FeatureSet-like objects.

# Description

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

#### Methods

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

46 ocSamples

		-	
$\sim$	าวก	n l	1/
UCI	Lap	LΥ	∟ y

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

ocSamples

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

```
ocSamples(n)
ocProbesets(n)
```

oligoSet 47

## **Arguments**

n

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

#### **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</pre>
```

oligoSnpSet-methods

Methods for oligoSnpSet class

## Description

Methods for oligoSnpSet class

### Methods

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

new("oligoSnpSet", ...): Instantiates an object of class oligoSnpSet. The assayData elements of the oligoSnpSet 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 oligoSnpSet object. Validity methods defined for the class will fail if the matrices are not integers. See examples for additional details.

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.

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.

parStatus 49

parStatus

Checks if oligo/crlmm can use parallel resources.

## **Description**

Checks if oligo/crlmm can use parallel resources (needs ff and snow package, in addition to options(cluster=makeCluster(...)).

# 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 51

## **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

Helper function to load packages.

# Description

This function checkes 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

52 requireClusterPkgSet

## **Examples**

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

 ${\tt requireClusterPkgSet} \quad \textit{DEPRECATED FUNCTIONS. Package loaders for clusters}.$ 

# 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.

 $\label{lem:continuous} \mbox{require For *ONE* package on the cluster nodes and accepts every argument taken by require.}$ 

# Value

Logical.

# Author(s)

Benilton S Carvalho

## See Also

require

sampleNames-methods 53

	C 1 C	T ( C ( 1.1	7
sampleNames-methods	Sample names for	r 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()
```

#### **Arguments**

... arguments to be passed to makeCluster in the 'snow' package.

54 sfsExample

#### **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)
```

SnpSet-methods 55

SnpSet-methods	Accessors and methods for SnpSet objects	

## **Description**

Utility functions for accessing data in SnpSet objects.

#### Usage

```
calls(object)
calls(object) <- value
confs(object, transform=TRUE)
confs(object) <- value</pre>
```

## **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:

```
round(-1000*log2(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 order and checkOrder.

56 SnpSet2-class

#### **Examples**

SnpSet2-class

Class "SnpSet2"

# **Description**

A container for genotype calls and confidence scores. Similar to the SnpSet class in **Biobase**, but SnpSet2 extends gSet directly whereas SnpSet extends eSet. Useful properties of 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**

```
genome: Object of class "character" indicating the UCSC genome build. Supported builds are
    'hg18' and 'hg19'.
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 "gSet", directly. Class "eSet", by class "gSet", distance 2. Class "VersionedBiobase", by class "gSet", distance 3. Class "Versioned", by class "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.</pre>
```

SnpSuperSet-class 57

## 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" ~~
```

# **Extends**

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

# Methods

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

## Author(s)

R. Scharpf

# See Also

AlleleSet

## **Examples**

```
showClass("SnpSuperSet")
## empty object from the class
x <- new("matrix")
new("SnpSuperSet", alleleA=x, alleleB=x, call=x, callProbability=x)</pre>
```

splitIndicesByLength Tools to distribute objects across nodes or by length.

## **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).
```

# Value

List.

# Author(s)

Benilton S Carvalho

#### See Also

split

# **Examples**

```
x <- 1:100
splitIndicesByLength(x, 8)
splitIndicesByLength(x, 8, balance=TRUE)
splitIndicesByNode(x)</pre>
```

sqsExample 59

sqsExample

SnpQSet Example

## **Description**

Example of SnpQSet instance.

# Usage

```
data(sqsExample)
```

#### **Format**

Belongs to SnpQSet class.

# **Examples**

```
data(sqsExample)
class(sqsExample)
```

SummarizedExperiment-methods

Methods for RangedSummarizedExperiment objects

# Description

Methods for RangedSummarizedExperiment.

# Usage

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

# Arguments

```
object A RangedSummarizedExperiment object.
... ignored
```

## **Details**

baf and 1rr are accessors for the B allele frequencies and log R ratio assays (matrices or arrays), respectively,

chromosome returns the seqnames of the rowRanges.

isSnp returns a logical vector for each marker in rowRanges indicating whether the marker targets a SNP (nonpolymorphic regions are FALSE).

## See Also

 ${\tt RangedSummarizedExperiment}$ 

# Index

* IO	chromosome2integer, 15
list.celfiles, 43	CopyNumberSet-methods, 18
* attribute	createFF, 20
getSequenceLengths, 32	featureDataList-methods, 22
* classes	fileConnections, 25
AlleleSet-class, 4	flags, 26
AssayData-methods, 5	genomeBuild, 29
BeadStudioSet-class, 9	geometry, 30
BeadStudioSetList-class, 10	getA, 30
CNSet-class, 15	getBar, 31
CopyNumberSet-class, 17	i2p, 36
DBPDInfo-class, 21	initializeBigMatrix, 37
FeatureSet-class, 23	integerMatrix, 38
ff_matrix-class, 24	is.ffmatrix, 39
ff_or_matrix-class, 25	isPackageLoaded, 39
ffdf-class, 24	kind, 40
gSet-class, 34	ldSetOptions, 41
gSetList-class, 35	library2, 42
ListClasses, 43	makeFeatureGRanges, 44
SnpSet2-class, 56	ocLapply, 46
SnpSuperSet-class, 57 * datasets	ocSamples, 46
	<pre>parStatus, 49 requireClusterPkgSet, 52</pre>
efsExample, 22 locusLevelData, 44	setCluster, 53
oligoSet, 47	SnpSet-methods, 55
scqsExample, 53	splitIndicesByLength, 58
sfsExample, 54	* methods
sqsExample, 59	assayDataList-methods,7
* data	batch, 7
pdPkgFromBioC, 49	batchStatistics, 8
requireAnnotation, 51	chromosome-methods, 14
* internal	CopyNumberSet-methods, 18
Deprecated, 21	db, 20
* list	exprs-methods, 22
affyPlatforms, 3	featureDataList-methods, 22
* manip	flags, 26
AssayDataList, 6	GenomeAnnotatedDataFrameFrom-methods,
assayDataList-methods, 7	28
batchStatistics, 8	GRanges-methods, 33
celfileDate, 11	isSnp-methods, 40
celfileName, 11	length-methods, 42
checkExists, 12	manufacturer-methods, 45
checkOrder, 13	oligoSnpSet-methods,48

platform-methods, 50	AffyTilingPDInfo-class
pmFragmentLength-methods, 50	(DBPDInfo-class), 21
position-methods, 51	allele (AlleleSet-class), 4
sampleNames-methods, 53	allele,AlleleSet-method
SummarizedExperiment-methods, 59	(AlleleSet-class), 4
* misc	allele, CNSet-method (CNSet-class), 15
affyPlatforms, 3	allele,SnpFeatureSet-method
generics, 27	(AlleleSet-class), 4
* utilities	AlleleSet, 57
list.celfiles, 43	AlleleSet (AlleleSet-class), 4
[,CNSet,ANY-method(CNSet-class), 15	AlleleSet-class, 4
[,CNSet-method(CNSet-class), 15	AnnotatedDataFrame, 27
[,gSetList,ANY-method(gSetList-class),	<pre>annotatedDataFrameFrom,ff_matrix-method</pre>
35	(ff_matrix-class), 24
[,gSetList-method(gSetList-class), 35	annotation, DBPDInfo-method
[[,BafLrrSetList,ANY,ANY-method	(DBPDInfo-class), 21
(BeadStudioSetList-class), 10	annotation,gSetList-method
[[,BeadStudioSetList,ANY,ANY-method	(gSetList-class), 35
(BeadStudioSetList-class), 10	annotationPackages, 5
[[<-,BafLrrSetList,ANY,ANY,BafLrrSet-method	AssayData, 28
(BeadStudioSetList-class), 10	AssayData-methods, 5
[[<-,gSetList,ANY,ANY,BafLrrSet-method	AssayDataList, 6
(gSetList-class), 35	assayDataList(assayDataList-methods), 7
\$,gSetList class), 35 \$,gSetList-method (gSetList-class), 35	assayDataList,gSetList-method
\$<-,gSetList-method(gSetList-class), 35	(gSetList-class), 35
5x-,gsetL1st-method (gsetL1st-Class), 35	assayDataList,oligoSetList-method
A (math) 20	(assayDataList-methods), 7
A (getA), 30	assayDataList-methods, 7
A, AlleleSet-method (getA), 30	assayDataNew, 7
A, CNSet-method (CNSet-class), 15	
A<- (getA), 30	B (getA), 30
A<-, AlleleSet, matrix-method (getA), 30	B, AlleleSet-method (getA), 30
A<-, AlleleSet-method (getA), 30	B, CNSet-method (CNSet-class), 15
A<-, CNSet-method (CNSet-class), 15	B<- (getA), 30
AffyExonPDInfo (DBPDInfo-class), 21	B<-,AlleleSet,matrix-method(getA),30
AffyExonPDInfo-class (DBPDInfo-class),	B<-, AlleleSet-method (getA), 30
21	B<-, CNSet-method (CNSet-class), 15
AffyExpressionPDInfo (DBPDInfo-class),	baf (generics), 27
21	baf,BafLrrSetList-method
AffyExpressionPDInfo-class	(BeadStudioSetList-class), 10
(DBPDInfo-class), 21	baf,BeadStudioSet-method
AffyGenePDInfo (DBPDInfo-class), 21	(BeadStudioSet-class), 9
AffyGenePDInfo-class (DBPDInfo-class),	baf,BeadStudioSetList-method
21	(BeadStudioSetList-class), 10
affyPlatforms, 3	baf,oligoSetList-method
AffySNPCNVPDInfo(DBPDInfo-class), 21	(BeadStudioSetList-class), 10
AffySNPCNVPDInfo-class	baf,oligoSnpSet-method
(DBPDInfo-class), 21	(oligoSnpSet-methods), 48
AffySNPPDInfo (DBPDInfo-class), 21	baf, Ranged Summarized Experiment-method
AffySNPPDInfo-class(DBPDInfo-class), 21	(SummarizedExperiment-methods),
AffySTPDInfo(DBPDInfo-class), 21	59
AffySTPDInfo-class(DBPDInfo-class), 21	baf,SummarizedExperiment-method
AffyTilingPDInfo(DBPDInfo-class), 21	(SummarizedExperiment-methods),

59	calls<-,SnpSet2,matrix-method
<pre>baf&lt;- (BeadStudioSet-class), 9</pre>	(SnpSet2-class), 56
baf<-,BeadStudioSet-method	callsConfidence,oligoSnpSet-method
(BeadStudioSet-class), 9	(oligoSnpSet-methods), 48
baf<-,oligoSnpSet-method	callsConfidence<-,oligoSnpSet,matrix-method
(oligoSnpSet-methods), 48	(oligoSnpSet-methods), 48
BafLrrSet-class (BeadStudioSet-class), 9	celfileDate, 11
BafLrrSetList-class	celfileName, 11
(BeadStudioSetList-class), 10	checkExists, 12
batch, 7, 8	check0rder, 13, 55
batch, CNSet-method (CNSet-class), 15	checkOrder,CopyNumberSet-method
batchNames, 8	(CopyNumberSet-class), 17
batchNames (batch), 7	<pre>checkOrder,gSet-method(gSet-class), 34</pre>
batchNames, AssayData-method	checkOrder,SnpSet-method
(AssayData-methods), 5	(SnpSet-methods), 55
batchNames, CNSet-method (CNSet-class),	chromosome, 35
batchNames<- (batch), 7	chromosome (chromosome-methods), 14
batchNames<-, AssayData-method	chromosome, AnnotatedDataFrame-method
(AssayData-methods), 5	(chromosome-methods), 14
batchNames<-, CNSet-method	chromosome,GenomeAnnotatedDataFrame-method
(CNSet-class), 15	(chromosome-methods), 14
batchStatistics, 8, 26	chromosome, GRanges-method
batchStatistics, CNSet-method	(chromosome-methods), 14
(CNSet-class), 15	chromosome, GRangesList-method
batchStatistics<- (batchStatistics), 8	(chromosome-methods), 14
batchStatistics<-,CNSet,AssayData-method	chromosome,gSet-method
(CNSet-class), 15	(chromosome-methods), 14
BeadStudioSet (BeadStudioSet-class), 9	chromosome,gSetList-method
BeadStudioSet-class, 9	(gSetList-class), 35
BeadStudioSetList, 36	chromosome, RangedDataCNV-method
BeadStudioSetList-class, 10	(Deprecated), 21
bothStrands (AlleleSet-class), 4	chromosome, RangedSummarizedExperiment-method
bothStrands,AlleleSet-method	(SummarizedExperiment-methods),
(AlleleSet-class), 4	59
bothStrands,SnpFeatureSet-method	chromosome, SnpSet-method
(AlleleSet-class), 4	(chromosome-methods), 14
	chromosome, SummarizedExperiment-method
calls (SnpSet-methods), 55	(SummarizedExperiment-methods), 59
calls, CNSet-method (CNSet-class), 15	chromosome-methods, 14
calls,oligoSetList-method	chromosome2integer, 14, 15
(BeadStudioSetList-class), 10	chromosome<- (chromosome-methods), 14
calls,oligoSnpSet-method	chromosome<-, GenomeAnnotatedDataFrame, integer-metho
(oligoSnpSet-methods), 48	(chromosome-methods), 14
calls, SnpSet-method (SnpSet-methods), 55	chromosome<-,gSet,integer-method
calls, SnpSet2-method (SnpSet2-class), 56	(chromosome-methods), 14
calls<- (SnpSet-methods), 55	chromosome<-,SnpSet,integer-method
<pre>calls&lt;-,CNSet,matrix-method   (CNSet-class), 15</pre>	(chromosome-methods), 14
calls<-,oligoSnpSet,matrix-method	chromosomePositionOrder (checkOrder), 13
(oligoSnpSet-methods), 48	clone2 (BeadStudioSetList-class), 10
calls<-,SnpSet,matrix-method	clone2,BafLrrSetList-method
(SnpSet-methods), 55	(BeadStudioSetList-class), 10
(5,55551535), 55	(2000), 10

close (fileConnections), 25	confs<-,SnpSet2,matrix-method
close, AlleleSet-method (getA), 30	(SnpSet2-class), 56
close, array-method (fileConnections), 25	copyNumber (CopyNumberSet-methods), 18
close, CNSet-method (CNSet-class), 15	copyNumber,BeadStudioSet-method
close, matrix-method (fileConnections),	(BeadStudioSet-class), 9
25	copyNumber,CopyNumberSet-method
close, numeric-method (fileConnections),	(CopyNumberSet-class), 17
25	copyNumber,oligoSetList-method
closeff(fileConnections), 25	(BeadStudioSetList-class), 10
closeff, CNSet-method (fileConnections),	copyNumber,oligoSnpSet-method
25	(oligoSnpSet-methods), 48
cnConfidence (CopyNumberSet-methods), 18	copyNumber<- (CopyNumberSet-methods), 18
cnConfidence,CopyNumberSet-method	copyNumber<-,BeadStudioSet,ANY-method
(CopyNumberSet-class), 17	(BeadStudioSet-class), 9
cnConfidence,oligoSnpSet-method	copyNumber<-,CopyNumberSet,matrix-method
(oligoSnpSet-methods), 48	(CopyNumberSet-class), 17
cnConfidence<- (CopyNumberSet-methods),	copyNumber<-,oligoSnpSet,matrix-method
18	(oligoSnpSet-methods), 48
cnConfidence<-,CopyNumberSet,matrix-method	CopyNumberSet (CopyNumberSet-class), 17
(CopyNumberSet-class), 17	CopyNumberSet-class, 17
cnConfidence<-,oligoSnpSet,matrix-method	CopyNumberSet-methods, 18
	corr (AssayData-methods), 5
(oligoSnpSet-methods), 48	corr, CNSet, character-method
CNSet (CNSet elece) 15	(CNSet-class), 15
CNSet (CNSet-class), 15	coverage2 (GRanges-methods), 33
CNSet-class, 15 coerce, AnnotatedDataFrame, GenomeAnnotatedDa	
coerce, Annotateduatarrame, Genomeannotatedua	(GRanges-methods), 33
(GenomeAnnotatedDataFrame-class),	coverage2,GRangesList-method
27	(GRanges-methods), 33
coerce, BeadStudioSet, data. frame-method	coverage2,RangedDataCNV-method
(BeadStudioSet-class), 9	(Deprecated), 21
coerce, CNSet, CopyNumberSet-method	createFF, 20
(CNSet-class), 15	,
coerce, CNSet, oligoSnpSet (CNSet-class),	db, 20
15	<pre>db,AlleleSet-method(AlleleSet-class),4</pre>
coerce, CNSet, oligoSnpSet-method	db, DBPDInfo-method (db), 20
(CNSet-class), 15	db, FeatureSet-method (db), 20
coerce, CNSetLM, CNSet-method	db,gSet-method(gSet-class),34
(CNSet-class), 15	db, SnpCnvQSet-method (db), 20
coerce,gSetList,list-method	db, SnpQSet-method (db), 20
(gSetList-class), 35	db, SnpSet-method (db), 20
coerce,oligoSnpSet,data.frame-method	db-methods (db), 20
(oligoSnpSet-methods), 48	DBPDInfo (DBPDInfo-class), 21
confs, 37	DBPDInfo-class, 21
confs (SnpSet-methods), 55	delCluster (setCluster), 53
confs, CNSet-method (CNSet-class), 15	delCluster-deprecated (setCluster), 53
confs, SnpSet-method (SnpSet-methods), 55	Deprecated, 21
confs, SnpSet2-method (SnpSet2-class), 56	<pre>dims,gSetList-method(gSetList-class),</pre>
confs<- (SnpSet-methods), 55	35
confs<-,CNSet,matrix-method	
(CNSet-class), 15	efsExample, 22
confs<-,SnpSet,matrix-method	elementNROWS,gSetList-method
(SnpSet-methods), 55	(gSetList-class), 35

eSet, 4, 9, 17, 23, 34, 56, 57	GeneFeatureSet (FeatureSet-class), 23
ExonFeatureSet (FeatureSet-class), 23	GeneFeatureSet-class
ExonFeatureSet-class	(FeatureSet-class), 23
(FeatureSet-class), 23	GenericFeatureSet (FeatureSet-class), 23
ExpressionFeatureSet	GenericFeatureSet-class
(FeatureSet-class), 23	(FeatureSet-class), 23
ExpressionFeatureSet-class	GenericPDInfo (DBPDInfo-class), 21
(FeatureSet-class), 23	GenericPDInfo-class (DBPDInfo-class), 21
ExpressionPDInfo(DBPDInfo-class), 21	generics, 27
ExpressionPDInfo-class	GenomeAnnotatedDataFrame, 28, 45
(DBPDInfo-class), 21	GenomeAnnotatedDataFrame
exprs, FeatureSet-method	(GenomeAnnotatedDataFrame-class),
(exprs-methods), 22	27
exprs, SnpSet2-method (SnpSet2-class), 56	GenomeAnnotatedDataFrame-class, 27
exprs-methods, 22	GenomeAnnotatedDataFrameFrom
•	
featureDataList	(GenomeAnnotatedDataFrameFrom-methods),
(featureDataList-methods), 22	28
featureDataList,gSetList-method	GenomeAnnotatedDataFrameFrom, array-method
(gSetList-class), 35	(GenomeAnnotatedDataFrameFrom-methods),
featureDataList-methods, 22	28
FeatureSet (FeatureSet-class), 23	GenomeAnnotatedDataFrameFrom, AssayData-method
FeatureSet-class, 23	(GenomeAnnotatedDataFrameFrom-methods),
featuresInRange (Deprecated), 21	28
featuresInRange,SnpSet2,RangedDataCNV-method	${\tt GenomeAnnotatedDataFrameFrom, ff\_or\_matrix-method}$
(Deprecated), 21	$({\tt GenomeAnnotatedDataFrameFrom-methods}),$
ff, 25	28
ff_matrix-class, 24	GenomeAnnotatedDataFrameFrom,list-method
ff_or_matrix-class, 25	$({\tt GenomeAnnotatedDataFrameFrom-methods}),$
ffdf, 25	28
ffdf-class, 24	GenomeAnnotatedDataFrameFrom,NULL-method
fileConnections, 25	$({\tt GenomeAnnotatedDataFrameFrom-methods}),$
findOverlans 45	28
findOverlaps,AnnotatedDataFrame,RangedDataCN	<u>√GenomeA</u> nnotatedDataFrameFrom-methods,
(Deprecated), 21	28
findOverlaps, GRanges, gSet-method	genomeBuild, 29
(GRanges-methods), 33	genomeBuild,DBPDInfo-method
	(genomeBuild), 29
findOverlaps, GRangesList, gSet-method	genomeBuild,FeatureSet-method
$({\tt GRanges-methods}), 33 \\ {\tt findOverlaps,RangedDataCNV,AnnotatedDataFrame} \\$	
Tindoveriaps, RangeduataCNV, AnnotateduataFram	genomeBuild,GRanges-method
(Deprecated), 21	(GRanges-methods), 33
findOverlaps, RangedDataCNV, CNSet-method	genomeBuild, gSet-method (gSet-class), 34
(Deprecated), 21	
findOverlaps,RangedDataCNV,RangedDataCNV-met	(gSetList-class), 35
(Deprecated), 21	
findOverlaps,RangedDataCNV,SnpSet-method	genomeBuild<- (genomeBuild), 29
(Deprecated), 21	genomeBuild<-,gSet,character-method
findOverlaps,RangedDataHMM,RangedDataHMM-met	
(Deprecated), 21	<pre>genomeBuild&lt;-,gSetList,character-method</pre>
flags, 26	(gSetList-class), 35
flags,AssayData-method	geometry, 30
(AssayData-methods), 5	geometry, DBPDInfo-method (geometry), 30
flags, CNSet-method (CNSet-class), 15	<pre>geometry, FeatureSet-method (geometry),</pre>

30	initialize,oligoSnpSet-method
getA, 30	(oligoSnpSet-methods), 48
getA,AlleleSet-method	initialize,SnpSet2-method
(AlleleSet-class), 4	(SnpSet2-class), 56
getA, SnpCnvQSet-method (getA), 30	initialize,SnpSuperSet-method
getA, SnpQSet-method (getA), 30	(SnpSuperSet-class), 57
getA, TilingFeatureSet2-method (getA), 30	initializeBigArray
getArm(gSet-class), 34	(initializeBigMatrix), 37
getArm, GenomeAnnotatedDataFrame-method	initializeBigMatrix,37
(GenomeAnnotatedDataFrame-class),	initializeBigVector
27	(initializeBigMatrix), 37
<pre>getArm, gSet-method (gSet-class), 34</pre>	integer2chromosome
getBar, 31	(chromosome2integer), 15
getCluster (setCluster), 53	integerArray(integerMatrix),38
getCluster-deprecated (setCluster), 53	integerMatrix, 38
getM (getA), 30	is.ffmatrix, 39
getM,AlleleSet-method	isPackageLoaded, 39
(AlleleSet-class), 4	isSnp, <i>35</i>
getM, SnpCnvQSet-method (getA), 30	isSnp(isSnp-methods), 40
getM, SnpQSet-method (getA), 30	<pre>isSnp,character-method(isSnp-methods),</pre>
getM, TilingFeatureSet2-method (getA), 30	40
getSequenceLengths, 32	isSnp,GenomeAnnotatedDataFrame-method
GRanges, 33, 45	(isSnp-methods), 40
GRanges-methods, 33	isSnp,gSet-method(isSnp-methods),40
gSet, 9, 56	isSnp,RangedSummarizedExperiment-method
gSet (gSet-class), 34	$({\sf SummarizedExperiment-methods}),$
gSet-class, 34	59
gSetList-class, 35	<pre>isSnp,SnpSet-method(isSnp-methods),40</pre>
800000000000000000000000000000000000000	isSnp,SummarizedExperiment-method
i2p, 36, 55	$({\sf SummarizedExperiment-methods}),$
	59
initialize, BeadStudioSet-method	isSnp-methods, 40
(BeadStudioSet-class), 9	
initialize, BeadStudioSetList-method	kind, $40$
(BeadStudioSetList-class), 10	kind, AffyExonPDInfo-method (kind), 40
<pre>initialize, CNSet-method (CNSet-class),</pre>	kind,AffyExpressionPDInfo-method
	(kind), 40
initialize, CNSetLM-method	kind, AffyGenePDInfo-method (kind), 40
(CNSet-class), 15	kind, AffyHTAPDInfo-method (kind), 40
initialize, CopyNumberSet-method	kind, AffySNPCNVPDInfo-method (kind), 40
(CopyNumberSet-class), 17	kind, AffySNPPDInfo-method (kind), 40
initialize, DBPDInfo-method	kind, ExpressionPDInfo-method (kind), 40
(DBPDInfo-class), 21	kind, FeatureSet-method (kind), 40
initialize, eSetList-method	kind, GenericPDInfo-method (kind), 40
(ListClasses), 43	kind, TilingPDInfo-method (kind), 40
initialize, FeatureSet-method	3 ID (1 /3 IC (0 () ) 41
(FeatureSet-class), 23	ldPath (ldSetOptions), 41
initialize, GenomeAnnotatedDataFrame-method	ldSetOptions, 41
(GenomeAnnotatedDataFrame-class),	ldStatus (ldSetOptions), 41
27	length, FeatureSet-method
initialize, gSet-method (gSet-class), 34	(length-methods), 42
<pre>initialize,gSetList-method   (gSetList-class), 35</pre>	length,gSetList-method
(goetlist-tidss), <del>jo</del>	(gSetList-class), 35

length-methods, 42	NgsExpressionPDInfo-class
library,42	(DBPDInfo-class), 21
library2,42	NgsTilingPDInfo (DBPDInfo-class), 21
list.celfiles, 43	NgsTilingPDInfo-class (DBPDInfo-class),
list.files,43	21
list_or_ffdf, 24	nu (AssayData-methods), 5
list_or_ffdf-class(ffdf-class), 24	nu,AssayData,character-method
ListClasses, 43	(AssayData-methods), 5
locusLevelData, 44, 48	nu, CNSet, character-method
lrr(generics), 27	(CNSet-class), 15
lrr,BafLrrSetList-method	numberProbes (GRanges-methods), 33
(BeadStudioSetList-class), 10	numberProbes, GRanges-method
lrr,BeadStudioSet-method	(GRanges-methods), 33
(BeadStudioSet-class), 9	numberProbes,GRangesList-method
lrr,BeadStudioSetList-method	(GRanges-methods), 33
(BeadStudioSetList-class), 10	( 1 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
lrr,RangedSummarizedExperiment-method	ocLapply, 46
(SummarizedExperiment-methods),	ocProbesets (ocSamples), 46
59	ocSamples, 46
lrr,SummarizedExperiment-method	oldClass, 24
(SummarizedExperiment-methods),	oligoSet, 47
59	oligoSetList, 36
lrr<-(BeadStudioSet-class),9	oligoSetList-class
lrr<-,BafLrrSet,ANY-method	(BeadStudioSetList-class), 10
(BeadStudioSet-class), 9	oligoSnpSet, 18
lrr<-,BafLrrSet-method	oligoSnpSet-class
(BeadStudioSet-class), 9	(oligoSnpSet-methods), 48
lrr<-,BafLrrSetList,matrix-method	oligoSnpSet-methods, 48
(BeadStudioSetList-class), 10	open (fileConnections), 25
lrr<-,BeadStudioSet,ANY-method	open, AlleleSet-method (getA), 30
(BeadStudioSet-class), 9	open, array-method (fileConnections), 25
lrr<-,BeadStudioSet-method	open, CNSet-method (CNSet-class), 15
(BeadStudioSet-class), 9	open, matrix-method (fileConnections), 25
(beaustudioset class), 9	open, numeric-method (fileConnections),
makeFeatureGRanges,44	25
makeFeatureGRanges,GenomeAnnotatedDataFrame	
(GenomeAnnotatedDataFrame-class),	openff, CNSet-method (fileConnections),
27	25
<pre>makeFeatureGRanges,gSet-method      (gSet-class), 34</pre>	order, <i>14</i> , <i>55</i>
makeFeatureGRanges,gSetList-method	p2i, 55
(gSetList-class), 35	p2i (i2p), 36
manufacturer (manufacturer-methods), 45	parStatus, 49
	pdPkgFromBioC, 49
manufacturer, DBPDInfo-method	phi (AssayData-methods), 5
(manufacturer-methods), 45	phi, AssayData, character-method
manufacturer, FeatureSet-method	(AssayData-methods), 5
(manufacturer-methods), 45	phi, CNSet, character-method
manufacturer-methods, 45	(CNSet-class), 15
matrix, 28	platform (platform-methods), 50
mean, RangedDataCBS-method (Deprecated),	
21	platform, FeatureSet-method
NgaEynnagaianDDInfa (DDDDInfa alasa) 21	(platform-methods), 50
NgsExpressionPDInfo(DBPDInfo-class), 21	platform-methods, 50

pmFragmentLength	sampleNames,gSetList-method
(pmFragmentLength-methods), 50	(gSetList-class), 35
pmFragmentLength,AffySNPPDInfo-method	sampleNames,RangedDataCNV-method
(pmFragmentLength-methods), 50	(Deprecated), 21
pmFragmentLength-methods, 50	sampleNames-methods, 53
position, 35	<pre>sampleNames&lt;-,gSetList,character-method</pre>
position (position-methods), 51	(gSetList-class), 35
position, AnnotatedDataFrame-method	sampleNames<-,RangedDataCNV,character-method
(position-methods), 51	(Deprecated), 21
position, GenomeAnnotatedDataFrame-method	scqsExample, 53
(position-methods), 51	se.exprs,FeatureSet-method
position,gSet-method	(exprs-methods), 22
(position-methods), 51	setCluster, 53
position,gSetList-method	setCluster-deprecated (setCluster), 53
(gSetList-class), 35	sfsExample, 54
position, SnpSet-method	show, BeadStudioSet-method
(position-methods), 51	(BeadStudioSet-class), 9
position-methods, 51	show, CNSet-method (CNSet-class), 15
position<-	show, DBPDInfo-method (DBPDInfo-class),
(GenomeAnnotatedDataFrame-class),	21
27	show, FeatureSet-method
<pre>position&lt;-,GenomeAnnotatedDataFrame,integer</pre>	-method (FeatureSet-class), 23
(GenomeAnnotatedDataFrame-class),	show, gSet-method (gSet-class), 34
27	show, gSetList-method (gSetList-class),
<pre>position&lt;-,oligoSnpSet,integer-method</pre>	35
(oligoSnpSet-methods), 48	
	sigma2, CNSet, character-method
RangedDataCBS (Deprecated), 21	(CNSet-class), 15
RangedDataCBS-class (Deprecated), 21	snpCallProbability, 55
RangedDataCNV (Deprecated), 21	<pre>snpCallProbability,CNSet-method</pre>
RangedDataCNV-class (Deprecated), 21	(CNSet-class), 15
RangedDataCopyNumber-class	SnpCnvFeatureSet (FeatureSet-class), 23
(Deprecated), 21	SnpCnvFeatureSet-class
RangedDataHMM (Deprecated), 21	(FeatureSet-class), 23
RangedDataHMM-class (Deprecated), 21	SNPCNVPDInfo (DBPDInfo-class), 21
RangedSummarizedExperiment, 59, 60	SNPCNVPDInfo-class (DBPDInfo-class), 21
read.celfiles, 23	SnpFeatureSet (FeatureSet-class), 23
read.xysfiles, 23	SnpFeatureSet-class (FeatureSet-class),
requireAnnotation, 51	23
requireClusterPkg	SNPPDInfo (DBPDInfo-class), 21
(requireClusterPkgSet), 52	SNPPDInfo-class (DBPDInfo-class), 21
requireClusterPkg-deprecated	snprma, 31
(requireClusterPkgSet), 52	SnpSet, <i>55</i> , <i>57</i>
requireClusterPkgSet, 52	SnpSet-methods, 55
requireClusterPkgSet-deprecated	SnpSet2-class, 56
(requireClusterPkgSet), 52	SnpSuperSet, 5
( • • • • • • • • • • • • • • • • • • •	<pre>SnpSuperSet (SnpSuperSet-class), 57</pre>
sampleNames, FeatureSet-method	SnpSuperSet-class, 57
(sampleNames-methods), 53	splitIndicesByLength, 58
sampleNames, GRanges-method	splitIndicesByNode
(GRanges-methods), 33	(splitIndicesByLength), 58
sampleNames, GRangesList-method	sqsExample, 59
(GRanges-methods), 33	state (GRanges-methods), 33

```
state, GRanges-method (GRanges-methods),
state,GRangesList-method
        (GRanges-methods), 33
state,RangedDataCNV-method
        (Deprecated), 21
SummarizedExperiment-methods, 59
tau2, CNSet, character-method
        (CNSet-class), 15
TilingFeatureSet (FeatureSet-class), 23
TilingFeatureSet-class
        (FeatureSet-class), 23
TilingFeatureSet2 (FeatureSet-class), 23
TilingFeatureSet2-class
        (FeatureSet-class), 23
TilingPDInfo (DBPDInfo-class), 21
TilingPDInfo-class (DBPDInfo-class), 21
updateObject,BeadStudioSet-method
        (BeadStudioSet-class), 9
update Object, Bead Studio Set List-method\\
        (BeadStudioSetList-class), 10
updateObject,CNSet-method
        (CNSet-class), 15
updateObject, GenomeAnnotatedDataFrame-method
        (GenomeAnnotatedDataFrame-class),
        27
updateObject,oligoSnpSet-method
        (oligoSnpSet-methods), 48
Versioned, 4, 9, 17, 23, 27, 34, 56, 57
VersionedBiobase, 4, 9, 17, 23, 34, 56, 57
```