Package ‘VanillaICE’

November 21, 2016

Version 1.36.0

Title A Hidden Markov Model for high throughput genotyping arrays

Author Robert Scharpf <rscharpf@jhu.edu>, Kevin Scharpf, and Ingo Ruczinski <ingo@jhsph.edu>

Maintainer Robert Scharpf <rscharpf@jhsph.edu>

Description Hidden Markov Models for characterizing chromosomal alterations in high throughput SNP arrays.

Date Sat Sep 27 11:46:03 EDT 2014

Depends R (>= 3.0.0), BiocGenerics (>= 0.13.6), GenomicRanges (>= 1.19.47), SummarizedExperiment (>= 0.2.0)

Imports Biobase, S4Vectors (>= 0.9.25), IRanges (>= 1.14.0), oligoClasses (>= 1.31.1), foreach, matrixStats, data.table, grid, lattice, methods, GenomeInfoDb, crlmm, tools, stats, utils, BSgenome.Hsapiens.UCSC.hg18

Suggests RUnit, SNPchip, human610quadv1bCrlmm, ArrayTV


Enhances doMC, doMPI, doSNOW, doParallel, doRedis

License LGPL-2

LazyLoad yes

biocViews CopyNumberVariation

Roxygen list(wrap=FALSE)

RoxygenNote 5.0.1

NeedsCompilation yes
### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>acf2</td>
<td>3</td>
</tr>
<tr>
<td>ArrayViews-class</td>
<td>3</td>
</tr>
<tr>
<td>baumWelchUpdate</td>
<td>6</td>
</tr>
<tr>
<td>calculateEmission</td>
<td>6</td>
</tr>
<tr>
<td>cnFilter</td>
<td>7</td>
</tr>
<tr>
<td>cn_means</td>
<td>8</td>
</tr>
<tr>
<td>CopyNumScanParams-class</td>
<td>11</td>
</tr>
<tr>
<td>doUpdate</td>
<td>12</td>
</tr>
<tr>
<td>dropDuplicatedMapLocs</td>
<td>13</td>
</tr>
<tr>
<td>dropSexChrom</td>
<td>13</td>
</tr>
<tr>
<td>emission</td>
<td>14</td>
</tr>
<tr>
<td>emissionParam</td>
<td>14</td>
</tr>
<tr>
<td>FilterParam-class</td>
<td>15</td>
</tr>
<tr>
<td>filters</td>
<td>16</td>
</tr>
<tr>
<td>genotypes</td>
<td>16</td>
</tr>
<tr>
<td>getExampleSnpExperiment</td>
<td>17</td>
</tr>
<tr>
<td>getHmmpParams</td>
<td>18</td>
</tr>
<tr>
<td>HMM-class</td>
<td>18</td>
</tr>
<tr>
<td>hmm2</td>
<td>19</td>
</tr>
<tr>
<td>HMMList</td>
<td>20</td>
</tr>
<tr>
<td>HMMList-class</td>
<td>21</td>
</tr>
<tr>
<td>HmmParam</td>
<td>22</td>
</tr>
<tr>
<td>hmmResults</td>
<td>23</td>
</tr>
<tr>
<td>HmmTrellisParam</td>
<td>23</td>
</tr>
<tr>
<td>IdiogramParams</td>
<td>23</td>
</tr>
<tr>
<td>IdiogramParams-class</td>
<td>24</td>
</tr>
<tr>
<td>isHeterozygous</td>
<td>25</td>
</tr>
<tr>
<td>LogLik</td>
<td>26</td>
</tr>
<tr>
<td>LogLik-class</td>
<td>26</td>
</tr>
<tr>
<td>lrrFile</td>
<td>27</td>
</tr>
<tr>
<td>matrixOrNULL</td>
<td>28</td>
</tr>
<tr>
<td>NA_filter</td>
<td>28</td>
</tr>
<tr>
<td>numberFeatures</td>
<td>28</td>
</tr>
<tr>
<td>parsedPath</td>
<td>29</td>
</tr>
<tr>
<td>parseSourceFile</td>
<td>29</td>
</tr>
<tr>
<td>probability</td>
<td>30</td>
</tr>
<tr>
<td>rescale</td>
<td>31</td>
</tr>
<tr>
<td>rowModes</td>
<td>31</td>
</tr>
<tr>
<td>segs</td>
<td>32</td>
</tr>
<tr>
<td>show,Viterbi-method</td>
<td>32</td>
</tr>
<tr>
<td>snpArrayAssays</td>
<td>33</td>
</tr>
<tr>
<td>SnpArrayExperiment-class</td>
<td>33</td>
</tr>
<tr>
<td>SnpExperiment</td>
<td>34</td>
</tr>
<tr>
<td>SnpGRanges-class</td>
<td>35</td>
</tr>
<tr>
<td>snp_exp</td>
<td>35</td>
</tr>
<tr>
<td>sourcePaths</td>
<td>36</td>
</tr>
<tr>
<td>start,oligoSnpSet-method</td>
<td>36</td>
</tr>
<tr>
<td>state,HmmGRanges-method</td>
<td>37</td>
</tr>
<tr>
<td>state-methods</td>
<td>37</td>
</tr>
<tr>
<td>sweepMode</td>
<td>37</td>
</tr>
</tbody>
</table>
acf2

Calculate lag10 autocorrelation

Description
A wrapper for the function acf that returns the autocorrelation for the specified lag. Missing values are removed.

Usage
acf2(x, lag = 10, ...)

Arguments
x
numeric vector
lag
integer
...
additional arguments to acf

See Also
acf

ArrayViews-class
ArrayViews class, constructor, and methods

Description
ArrayViews provides views to the low-level data – log R ratios, B allele frequencies, and genotypes that are stored in parsed files on disk, often scaled and coerced to an integer. Accessors to the low-level data are provided that extract the marker-level summaries from disk, rescaling when appropriate.
ArrayViews-class

Usage

ArrayViews(class = "ArrayViews", colData, rowRanges = GRanges(),
sourcePaths = character(), scale = 1000, sample_ids,
parsedPath = getwd(), lrrFiles = character(), bafFiles = character(),
gtfFiles = character())

## S4 method for signature 'ArrayViews,ANY,ANY,ANY'
x[1, j, ..., drop = FALSE]

colnames(x) <- value

## S4 method for signature 'ArrayViews'
colnames(x, do.NULL = TRUE, prefix = "col")

## S4 method for signature 'ArrayViews'
x$name

## S4 replacement method for signature 'ArrayViews'
x$name <- value

## S4 method for signature 'ArrayViews'
show(object)

## S4 method for signature 'ArrayViews'
sapply(X, FUN, ..., simplify = TRUE,
USE.NAMES = TRUE)

## S4 method for signature 'ArrayViews'
ncol(x)

## S4 method for signature 'ArrayViews'
nrow(x)

## S4 method for signature 'ArrayViews'
dim(x)

## S4 method for signature 'ArrayViews'
start(x)

Arguments

class character string
colData DataFrame
colRanges GRanges object
sourcePaths character string provide complete path to plain text source files (one file per
sample) containing log R ratios and B allele frequencies
scale log R ratios and B allele frequencies can be stored as integers on disk to increase IO speed. If scale =1, the raw data is not transformed. If scale = 1000 (default),
the log R ratios and BAFs are multiplied by 1000 and coerced to an integer.
sample_ids character vector indicating how to name samples. Ignored if colData is specified.
parsedPath character vector indicating where parsed files should be saved
ArrayViews-class

lrrFiles  character vector of file names for storing log R ratios
bafFiles  character vector of file names for storing BAFs
gtFiles  character vector of file names for storing genotypes
x  a ArrayViews object
i  numeric vector or missing
j  numeric vector or missing
...  additional arguments to FUN
drop  ignored
value  a character-string vector
do.NULL  ignored
prefix  ignored
name  character string indicating name in colData slot of ArrayViews object
object  a ArrayViews object
X  a ArrayViews object
FUN  a function to apply to each column of X
simplify  logical indicating whether result should be simplified
USE.NAMES  whether the output should be a named vector

Slots

colData  A character string
rowRanges  A DataFrame. WARNING: The accessor for this slot is rowRanges, not rowRanges!
index  A GRanges object
sourcePaths  A character string providing complete path to source files (one file per sample) containing low-level summaries (Log R ratios, B allele frequencies, genotypes)
scale  A length-one numeric vector
parsedPath  A character string providing full path to where parsed files should be saved
lrrFiles  A character string providing full path to where parsed files should be saved
bafFiles  character vector of filenames for log R ratios
gtFiles  character vector of filenames for genotypes

See Also

CopyNumScanParams parseSourceFile

Examples

ArrayViews()
## From unit test
require(BSgenome.Hsapiens.UCSC.hg18)
require(data.table)
extdir <- system.file("extdata", package="VanillaICE", mustWork=TRUE)
features <- suppressWarnings(fread(file.path(extdir, "SNP_info.csv")))
fggr <- GRanges(paste0("chr", features$Chr), IRanges(features$Position, width=1),
isSnp=features["Intensity Only"]==0)
fggr <- SnpGRanges(fggr)
names(fggr) <- features["Name"]
bsgenome <- BSgenome.Hsapiens.UCSC.hg18
seqlevels(fgr) <- seqlevels(bsgenome)[seqlevels(bsgenome) %in% seqlevels(fgr)]
seqinfo(fgr) <- seqinfo(bsgenome)[seqlevels(fgr),]

fgr <- sort(fgr)
files <- list.files(extdir, full.names=TRUE, recursive=TRUE, pattern="FinalReport")
ids <- gsub(".rds", "", gsub("FinalReport", "", basename(files)))
views <- ArrayViews(rowRanges=fgr,
  sourcePaths=files,
  sample_ids=ids)
lrrFile(views)

## view of first 10 markers and samples 3 and 5
views <- views[1:10, c(3,5)]

baumWelchUpdate Function for updating parameters for emission probabilities

Description
This function is not meant to be called directly by the user. It is exported in the package NAMESPACE for internal use by other BioC packages.

Usage
baumWelchUpdate(param, assay_list)

Arguments
param A container for the HMM parameters
assay_list list of log R ratios and B allele frequencies

calculateEmission Calculate the emission probabilities for the 6-state HMM

Description
Given the data and an object containing parameters for the HMM, this function computes emission probabilities. This function is not intended to be called by the user and is exported for internal use by other BioC packages.

Usage
calculateEmission(x, param = EmissionParam())

Arguments
x list of low-level data with two elements: a numeric vector of log R ratios and a numeric vector of B allele frequencies
param parameters for the 6-state HMM
**cnvFilter**

**Value**
A matrix of emission probabilities. Column correspond to the HMM states and rows correspond to markers on the array (SNPs and nonpolymorphic markers)

**See Also**
baumWelchUpdate

---

**cnvFilter** *Filter the HMM-derived genomic ranges for copy number variants*

**Description**
The HMM-derived genomic ranges are represented as a GRanges-derived object. cnvFilter returns a GRanges object using the filters stipulated in the filters argument.

**Usage**

cnvFilter(object, filters = FilterParam())
cnvSegs(object, filters = FilterParam(state = c("1", "2", "5", "6")))
duplication(object, filters = FilterParam(state = c("5", "6")))
deletion(object, filters = FilterParam(state = c("1", "2")))
hemizygous(object, filters = FilterParam(state = "2"))
homozygous(object, filters = FilterParam(state = "1"))

## S4 method for signature 'HMM'
cnvSegs(object, filters = FilterParam(state = as.character(c(1, 2, 5, 6))))

## S4 method for signature 'HMMList'
segs(object)

## S4 method for signature 'HMMList'
hemizygous(object)

## S4 method for signature 'HMMList'
homozygous(object)

## S4 method for signature 'HMMList'
duplication(object)

cnvSegs(object, filters = FilterParam(state = as.character(c(1, 2, 5, 6))))

## S4 method for signature 'HMMList'

## S4 method for signature 'HMMList'

## S4 method for signature 'HMMList'
cn_means

A parameter class for computing Emission probabilities

Description

Parameters for computing emission probabilities for a 6-state HMM, including starting values for
the mean and standard deviations for log R ratios (assumed to be Gaussian) and B allele frequencies
(truncated Gaussian), and initial state probabilities.

Constructor for EmissionParam class

This function is exported primarily for internal use by other BioC packages.

Usage

cn_means(object)

cn_sds(object)

baf_means(object)

baf_sds(object)

baf_means(object) <- value
baf_sds(object) <- value

cn_sds(object) <- value

cn_means(object) <- value

EmissionParam(cn_means = CN_MEANS(), cn_sds = CN_SDS(),
baf_means = BAF_MEANS(), baf_sds = BAF_SDS(), initial = rep(1/6, 6),
EMupdates = 5L, CN_range = c(-5, 3), temper = 1, p_outlier = 1/100,
modelHomozygousRegions = FALSE)

EMupdates(object)

## S4 method for signature 'EmissionParam'
show(object)

Arguments

object see showMethods("EMupdates")

value numeric vector

cn_means numeric vector of starting values for log R ratio means (order is by copy number state)

cn_sds numeric vector of starting values for log R ratio standard deviations (order is by copy number state)

baf_means numeric vector of starting values for BAF means ordered. See example for details on how these are ordered.

baf_sds numeric vector of starting values for BAF means ordered. See example for details on how these are ordered.

initial numeric vector of initial state probabilities

EMupdates number of EM updates

CN_range the allowable range of log R ratios. Log R ratios outside this range are thresholded.

temper Emission probabilities can be tempered by emit^temper. This is highly experimental.

p_outlier probability that an observation is an outlier (assumed to be the same for all markers)

modelHomozygousRegions logical. If FALSE (default), the emission probabilities for BAFs are modeled from a mixture of truncated normals and a Unif(0,1) where the mixture probabilities are given by the probability that the SNP is heterozygous. See Details below for a discussion of the implications.

Details

The log R ratios are assumed to be emitted from a normal distribution with a mean and standard deviation that depend on the latent copy number. Similarly, the BAFs are assumed to be emitted from a truncated normal distribution with a mean and standard deviation that depends on the latent number of B alleles relative to the total number of alleles (A+B).
Value
numeric vector

Details
When modelHomozygousRegions is FALSE (the default in versions >= 1.28.0), emission probabilities for B allele frequencies are calculated from a mixture of a truncated normal densities and a Unif(0,1) density with the mixture probabilities given by the probability that a SNP is homozygous. In particular, let $p$ denote a 6 dimensional vector of density estimates from a truncated normal distribution for the latent genotypes 'A', 'B', 'AB', 'AAB', 'ABB', 'AAAB', and 'ABBB'. The probability that a genotype is homozygous is estimated as

$$ pr_{Hom} = \frac{p["A"] + p["B"]}{\text{sum}(p)} $$

and the probability that the genotype is heterozygous (any latent genotype that is not 'A' or 'B') is given by

$$ pr_{Het} = 1 - pr_{Hom} $$

Since the density of a Unif(0,1) is 1, the 6-dimensional vector of emission probability at a SNP is given by

$$ emit = pr_{Het} * p + (1 - pr_{Het}) $$

The above has the effect of minimizing the influence of BAFs near 0 and 1 on the state path estimated by the Viterbi algorithm. In particular, the emission probability at homozygous SNPs will be virtually the same for states 3 and 4, but at heterozygous SNPs the emission probability for state 3 will be an order of magnitude greater for state 3 (diploid) compared to state 4 (diploid region of homozygosity). The advantage of this parameterization are fewer false positive hemizygous deletion calls. [ Log R ratios tend to be more sensitive to technical sources of variation than the corresponding BAFs/genotypes. Regions in which the log R ratios are low due to technical sources of variation will be less likely to be interpreted as evidence of copy number loss if heterozygous genotypes have more 'weight' in the emission estimates than homozygous genotypes. ] The trade-off is that only states estimated by the HMM are those with copy number alterations. In particular, copy-neutral regions of homozygosity will not be called.

By setting modelHomozygousRegions = TRUE, the emission probabilities at a SNP are given simply by the $p$ vector described above and copy-neutral regions of homozygosity will be called.

Examples

```r
ep <- EmissionParam()
cn_means(ep)
ep <- EmissionParam()
cn_sds(ep)
ep <- EmissionParam()
baf_means(ep)
ep <- EmissionParam()
baf_sds(ep)
ep <- EmissionParam()
baf_means(ep) <- baf_means(ep)
ep <- EmissionParam()
baf_sds(ep) <- baf_sds(ep)
```
ep <- EmissionParam()
cn_sds(ep) <- cn_sds(ep)
ep <- EmissionParam()
cn_means(ep) <- cn_means(ep)
ep <- EmissionParam()
show(ep)
cn_means(ep)
cn_sds(ep)
baf_means(ep)
baf_sds(ep)

---

CopyNumScanParams-class

Parameters for parsing source files containing SNP-array processed data, such as GenomeStudio files for the Illumina platform

Description

Raw SNP array processed files have headers and variable labels that may depend the software, how the output files was saved, the software version, and other factors. The purpose of this container is to collect the parameters relevant for reading in the source files for a particular project in a single container. This may require some experimentation as the example illustrates. The function fread in the data.table package greatly simplifies this process.

Usage

CopyNumScanParams(cnvar = "Log R Ratio", bafvar = "B Allele Freq", 
                    gtvar = c("Allele1 - AB", "Allele2 - AB"), index_genome = integer(), 
                    select = integer(), scale = 1000, row.names = 1L)

## S4 method for signature 'CopyNumScanParams'
show(object)

Arguments

- **cnvar**
  - length-one character vector providing name of variable for log R ratios

- **bafvar**
  - length-one character vector providing name of variable for B allele frequencies

- **gtvar**
  - length-one character vector providing name of variable for genotype calls

- **index_genome**
  - integer vector indicating which rows of the of the source files (e.g., GenomeStudio) to keep. By matching on a sorted GRanges object containing the feature annotation (see example), the information on the markers will also be sorted.

- **select**
  - integer vector specifying indicating which columns of the source files to import (see examples)

- **scale**
  - length-one numeric vector for rescaling the raw data and coercing to class integer. By default, the low-level data will be scaled and saved on disk as integers.

- **row.names**
  - length-one numeric vector indicating which column the SNP names are in

- **object**
  - a CopyNumScanParams object
doUpdate

Slots

index_genome an integer vector
cnvar the column label for the log R ratios
bafvar the column label for the B allele frequencies
gtvar the column label(s) for the genotypes
scale length-one numeric vector indicating how the low-level data should be scaled prior to saving on disk
select numeric vector indicating which columns to read
row.names length-one numeric vector indicating which column the SNP names are in

See Also

ArrayViews parseSourceFile

Examples

CopyNumScanParams() ## empty container

doUpdate(param)

Helper function to determine whether to update the HMM parameters via the Baum-Welch algorithm

Description

This function is not intended to be called directly by the user, and is exported only for internal use by other BioC packages.

Usage

doUpdate(param)

Arguments

param An object containing parameters for the HMM

See Also

HmmParam
**dropDuplicatedMapLocs**

*Drop markers on the same chromosome having the same genomic coordinates*

**Description**

If there are multiple markers on the same chromosome with the same annotated position, only the first is kept.

**Usage**

\[\text{dropDuplicatedMapLocs}(\text{object})\]

**Arguments**

- **object**: a container for which the methods seqnames and start are defined

**Value**

an object of the same class with duplicated genomic positions removed

**Examples**

```r
data(snp_exp)
g <- rowRanges(snp_exp)
## duplicate the first row
g[length(g)] <- g[1]
rowRanges(snp_exp) <- g
snp_exp2 <- dropDuplicatedMapLocs(snp_exp)
```

---

**dropSexChrom**

*Filter sex chromosomes*

**Description**

Removes markers on chromosomes X and Y.

**Usage**

\[\text{dropSexChrom}(\text{object})\]

**Arguments**

- **object**: an object for which the methods seqnames and rowRanges are defined.

**Value**

an object of the same class as the input
emission  
Methods to set and get emission probabilities

Description
Get or set a matrix of emission probabilities. This function is exported primarily for internal use by other BioC packages.

Usage
emission(object)
emission(object) <- value

Arguments
object see showMethods(emission)
value a matrix of emission probabilities

Value
matrix

emissionParam  
Accessor for parameters used to compute emission probabilities

Description
Parameters for computing emission probabilities include the starting values for the Baum Welch update and initial state probabilities.

Usage
emissionParam(object)
emissionParam(object) <- value

Arguments
object an object of class EmissionParam
value an object of class EmissionParam

Value
EmissionParam instance
Examples

```r
hparam <- HmmParam()
emissionParam(hparam)
ep <- EmissionParam()
  cn_means(ep) <- log2(c(.1/2, 1/2, 2/2, 2/2, 3/2, 4/2))
emissionParam(hparam) <- ep
```

Description

The maximum a posteriori estimate of the trio copy number state for each genomic range is represented in a `GRanges`-derived class. Ultimately, these ranges will be filtered based on the trio copy number state (e.g., denovo deletions), size, number of features (SNPs), or chromosome. `FilterParam` is a container for the parameters commonly used to filter the genomic ranges.

Usage

```r
FilterParam(probability = 0.99, numberFeatures = 10,
  seqnames = paste0("chr", c(1:22, "X", "Y")), state = as.character(1:6),
  width = 1L)
```

Arguments

- `probability` minimum probability for the call
- `numberFeatures` minimum number of SNPs/nonpolymorphic features in a region
- `seqnames` the seqnames (character string or `Rle` to keep)
- `state` character: the HMM states to keep
- `width` the minimum width of a region
- `object` a `FilterParam` object

Slots

- `probability` a length-one numeric vector indicating the minimum posterior probability for the called state. Genomic intervals with posterior probabilities below `probability` will be filtered.
- `numberFeatures` a positive integer indicating the minimum number of features in a segment
- `seqnames` a character vector of seqnames to select (i.e., 'chr1' for only those intervals on chromosome 1)
- `width` positive integer indicating the minimal width of genomic intervals
- `state` character string indicating which hidden Markov model states to select
See Also
cnvFilter cnvSegs hmm2

Examples

```r
fp <- FilterParam()
width(fp)
numberFeatures(fp)
seqnames(fp)
## To select CNV segments for which
## - the CNV call has a 'posterior' probability of at least 0.95
## - the number of features is at least 10
## - the HMM states are 1 (homozygous deletion) or 2 (hemizygous deletion)
FilterParam(probability=0.95, numberFeatures=10, state=c("1", "2"))
```

filters

Accessor for HMM filter parameters

Description

Accessor for HMM filter parameters

Usage

`filters(object)`

Arguments

- `object` see `showMethods(filters)`

---

genotypes

Accessor for SNP genotypes

Description

Extract SNP genotypes. Genotypes are assumed to be represented as integers: 1=AA, 2=AB, 3=BB.

Usage

`genotypes(object)`

## S4 method for signature 'ArrayViews'
`lrr(object)`

## S4 method for signature 'ArrayViews'
`baf(object)`

## S4 method for signature 'ArrayViews'
`genotypes(object)`
getExampleSnpExperiment

Create an example SnpArrayExperiment from source files containing marker-level genomic data that are provided in this package

Usage

getExampleSnpExperiment(bsgenome)

Arguments

bsgenome a BSgenome object

Value

A SnpArrayExperiment

Examples

## Not run:
if(require("BSgenome.Hsapiens.UCSC.hg18")){
  genome <- BSgenome.Hsapiens.UCSC.hg18
  snp_exp <- getExampleSnpExperiment(genome)
}

## End(Not run)
getHmmParams

Accessor for HMM model parameters

Description
Accessor for HMM model parameters

Usage
getHmmParams(object)

Arguments
object see showMethods(HmmParam)

Examples
hmm_object <- HMM()
getHmmParams(hmm_object)

HMM-class

Container for the segmented data and the 6-state HMM model parameters

Description
Container for the segmented data and the 6-state HMM model parameters
The constructor HMM creates an object of class HMM. Not typically called directly by the user.

Usage
HMM(granges = GRanges(), param = HmmParam(), posterior = matrix(),
    filters = FilterParam())

## S4 method for signature 'HMM'
state(object)

## S4 method for signature 'HMM'
show(object)

Arguments
granges a GRanges object
param a HmmParam object
posterior matrix of posterior probabilities
filters an object of class FilterParam
object a HMM object
hmm2

Slots

- granges a GRanges object
- param a HmmParam object
- posterior a matrix of posterior probabilities
- filters a FilterParam object

See Also

- hmm2

Examples

```r
data(snp_exp)
hmm_list <- hmm2(snp_exp[,1])
resultsFirstSample <- hmm_list[[1]]
resultsFirstSample
```

hmm2

Fit a 6-state HMM to log R ratios and B allele frequencies estimated from SNP arrays

Description

This function is intended for estimating the integer copy number from germline or DNA of clonal origin using a 6-state HMM. The states are homozygous deletion, hemizygous deletion, diploid copy number, diploid region of homozygosity, single copy gain, and two+ copy gain. Because heterozygous markers are more informative for copy number than homozygous markers and regions of homozygosity are common in normal genomes, we currently computed a weighted average of the BAF emission matrix with a uniform 0,1 distribution by the probability that the marker is heterozygous, thereby downweighting the contribution of homozygous SNPs to the likelihood. In addition to making the detection of copy-neutral regions of homozygosity less likely, it also helps prevent confusing hemizygous deletions with copy neutral regions of homozygosity – the former would be driven mostly by the log R ratios. This is experimental and subject to change.

Usage

```r
hmm2(object, emission_param = EmissionParam(),
     transition_param = TransitionParam(), ...)
```

## S4 method for signature 'SnpArrayExperiment'
```r
hmm2(object, emission_param = EmissionParam(),
     transition_param = TransitionParam(), ...)
```

## S4 method for signature 'oligoSnpSet'
```r
hmm2(object, emission_param = EmissionParam(),
     transition_param = TransitionParam(), ...)
```

## S4 method for signature 'ArrayViews'
```r
hmm2(object, emission_param = EmissionParam(),
     transition_param = TransitionParam(), tolerance = 2, verbose = FALSE,
     ...)
Arguments

- **object** A `SnpArrayExperiment`
- **emission_param** A `EmissionParam` object
- **transition_param** A `TransitionParam` object
- **tolerance** length-one numeric vector. When the difference in the log-likelihood of the Viterbi state path between successive models (updated by Baum Welch) is less than the tolerance, no additional model updates are performed.
- **verbose** logical. Whether to display messages indicating progress.

Details

The `hmm2` method allows parallelization across samples using the foreach paradigm. Parallelization is automatic when enabled via packages such as snow/doSNOW.

Examples

```r
tp <- TransitionParam()
TransitionParam(taup=1e12)
data(snp_exp)
emission_param <- EmissionParam(temper=1/2)
fit <- hmm2(snp_exp, emission_param)
unlist(fit)
cnvSegs(fit)
```

```r
## There is too little data to infer cnv reliably in this trivial example.
## To illustrate filtering options on the results, we select CNVs for which
## - the CNV call has a posterior probability of at least 0.5
## - the number of features is 2 or more
## - the HMM states are 1 (homozygous deletion) or 2 (hemizygous deletion)
fp <- FilterParam(probability=0.5, numberFeatures=2, state=c("1", "2"))
cnvSegs(fit, fp)
```

```r
## for parallelization
## Not run:
library(snow)
library(doSNOW)
cl <- makeCluster(2, type = "SOCK")
registerDoSNOW(cl)
fit <- hmm2(snp_exp, emission_param)
```

## End(Not run)

**HMMList**

Constructor for HMMList class

Description

The constructor function for the HMMList class. The constructor is useful for representing a list of HMM objects.
Usage

HMMList(object)

Arguments

object a list. Each element of the list is in instance of the HMM class.

See Also

HMMList HMM hmm2

Description

Each element of the HMMList contains the genomic intervals of the HMM segmentation (GRanges-derived object), parameters from the Baum-Welch, and a FilterParam object.

Usage

## S4 method for signature 'HMMList'
show(object)

## S4 method for signature 'HMMList'
unlist(x, recursive = TRUE, use.names = TRUE)

Arguments

object a HMMList object
x a HMMList object
recursive logical; currently ignored
use.names logical; currently ignored

Slots

.Data a list. Each element of the list should be a HMM object.

See Also

HMM

Examples

data(snp_exp)
fit <- hmm2(snp_exp)
class(fit)
identical(length(fit), ncol(snp_exp))
unlist(fit)
**HmmParam**

Constructor for HmmParam class

**Description**

Contains emission probabilities, parameters for emission probabilities, and transition probabilities required for computing the most likely state path via the Viterbi algorithm.

**Usage**

```r
HmmParam(emission = matrix(0, 0, 0), emission_param = EmissionParam(),
  transition = rep(0.99, nrow(emission)),
  chromosome = character(nrow(emission)), loglik = LogLik(),
  viterbi = Viterbi(), compute_posteriors = TRUE, verbose = FALSE)
```

```r
## S4 method for signature 'HmmParam'
show(object)
```

```r
## S4 method for signature 'HmmParam'
nrow(x)
```

```r
## S4 method for signature 'HmmParam'
ncol(x)
```

**Arguments**

- `emission`: A matrix of emission probabilities
- `emission_param`: an object of class `EmissionParam`
- `transition`: vector of transition probabilities whose length is N-1, where N is the number of markers. User should provide the probability that the state at marker j is the same as the state at marker j-1. It is assumed that the probability of transitioning to state_j from state_j-1 is the same for all states != state_j-1.
- `chromosome`: character vector
- `loglik`: an object of class `LogLik`
- `viterbi`: an object of class `Viterbi`
- `compute_posteriors`: logical
- `verbose`: logical
- `object`: a HmmParam object
- `x`: a HmmParam object

**Examples**

```r
HmmParam()
```
**hmmResults**

*Example output from the hidden markov model*

**Description**

The results of a 6-state HMM fit to simulated copy number and genotype data.

**Format**

- a GRanges object

**HmmTrellisParam**  
*Constructor for HmmTrellisParam class*

**Description**

Constructor for HmmTrellisParam class

**Usage**

```r
HmmTrellisParam(ylimits = list(c(0, 1), c(-3, 1)), expandfun = function(g) {
    width(g) * 50 })
```

**Arguments**

- `ylimits`: length-two list of the y-axis limits for B allele frequencies and log R ratios, respectively
- `expandfun`: a function that takes a length-one GRanges object as an argument and computes a width relative to the width of the GRanges object

**IdiogramParams**  
*Constructor for IdiogramParam objects*

**Description**

Parameters for plotting idiograms

**Usage**

```r
IdiogramParams(seqnames = character(), seqlengths = numeric(),
    unit = "kb", genome = "hg19", box = list(color = "blue", lwd = 1))
```

```r
## S4 method for signature 'IdiogramParams,ANY'
plot(x, y, ...)
```
Arguments

seqnames  length-one character vector providing chromosome name
seqlengths length-one numeric vector indicating size of chromosome
unit       character string indicating unit for genomic position
genome     character string indicating genome build
box        a list of parameters for plotting the box around the part of the idiogram that is plotted
x          an IdiogramParam object
y          ignored
...        ignored

Value

IdiogramParam object

Description

Parameter class for plotting idiograms

Usage

```r
## S4 method for signature 'IdiogramParams'
show(object)
```

Arguments

object an IdiogramParam object

Slots

seqnames length-one character vector providing chromosome name
seqlengths length-one numeric vector indicating size of chromosome
unit character string indicating unit for genomic position (default is 'kb')
genome character string indicating genome build
box a list of parameters for plotting the box around the part of the idiogram that is plotted.
isHeterozygous

Assess whether genotype is heterozygous based on BAFs

Usage

isHeterozygous(object, cutoff)

## S4 method for signature 'ArrayViews'
isHeterozygous(object, cutoff)

## S4 method for signature 'SnpArrayExperiment'
isHeterozygous(object, cutoff)

## S4 method for signature 'numeric'
isHeterozygous(object, cutoff)

## S4 method for signature 'matrix'
isHeterozygous(object, cutoff)

Arguments

object a SnpArrayExperiment or ArrayViews object containing BAFs, a matrix of BAFs, or a numeric vector of BAFs. vector of BAFs

cutoff a length-two numeric vector providing the range of BAFs consistent with allelic heterozygosity

Examples

if(require("BSgenome.Hsapiens.UCSC.hg18")){
  bsgenome <- BSgenome.Hsapiens.UCSC.hg18
  snp_exp <- getExampleSnpExperiment(bsgenome)
LogLik-class

is_het <- isHeterozygous(snp_exp[, 1], c(0.4, 0.6))
table(is_het)
}

LogLik Constructor for LogLik class

Description
A container for the log likelihood of the Viterbi state path. Stores the log likelihood from successive updates of model parameters. When the difference between the log likelihoods at iteration i and i-1 is below the tolerance, no additional updates are performed.

Usage
LogLik(loglik = numeric(), tolerance = 1L)

Arguments
loglik length-one numeric vector for the log likelihood of the Viterbi state path
tolerance if the difference in the log-likelihood of the Viterbi state path after the Baum-Welch update is less than the specified tolerance, no additional Baum-Welch updates are required

See Also
LogLik

LogLik-class Classes and methods for storing/getting log-likelihoods from Viterbi algorithm

Description
Exported for internal use by other BioC packages

Usage
## S4 method for signature 'LogLik'
length(x)

## S4 method for signature 'LogLik'
show(object)

Arguments
x object of class LogLik
object a LogLik object
Slots

loglik a numeric vector
tolerance a numeric vector

See Also
LogLik

Description
Accessors for objects of class ArrayViews

Usage
lrrFile(object)
lrrFile(object) <- value
bafFile(object)
gtFile(object)

## S4 method for signature 'ArrayViews'
lrrFile(object)

## S4 replacement method for signature 'ArrayViews'
lrrFile(object) <- value

## S4 method for signature 'ArrayViews'
bafFile(object)

## S4 method for signature 'ArrayViews'
gtFile(object)

Arguments

object see showMethods("lrrFile")
value a character vector of filenames for the log R ratios

Examples
views <- ArrayViews(parsedPath=tempdir())
sourcePaths(views)
lrrFile(views)
bafFile(views)
gtFile(views)
**matrixOrNULL**

**A class allowing matrix or NULL objects**

**Description**

Exported for internal use by other BioC packages

**NA_filter**

**Remove SNPs with NAs in any of the low-level estimates**

**Description**

Remove SNPs with NAs in any of the low-level estimates

**Usage**

`NA_filter(x, i)`

**Arguments**

- `x` a container for SNP data (`SnpArrayExperiment`)
- `i` integer vector to subset

**Value**

An object of the same class

**numberFeatures**

**The number of SNP/nonpolymorphic probes contained in a genomic interval**

**Description**

The number of SNP/nonpolymorphic probes contained in a genomic interval

**Usage**

`numberFeatures(object)`

**Arguments**

- `object` see `showMethods(numberFeatures)`
parsedPath

Complete path to directory for keeping parsed files

Description
A character string indicating the complete path for storing parsed files.

Usage
parsedPath(object)

## S4 method for signature 'ArrayViews'
parsedPath(object)

Arguments
object a ArrayViews object

See Also
parseSourceFile ArrayViews
ArrayViews

parseSourceFile Function for parsing GenomeStudio files

Description
This function parses genome studio files, writing the low-level data for log R ratios, B allele frequencies, and genotypes to disk as integers (1 file per subject per data type).

Usage
parseSourceFile(object, param)

## S4 method for signature 'ArrayViews,CopyNumScanParams'
parseSourceFile(object, param)

Arguments
object An ArrayViews object
param An object of class CopyNumScanParams

See Also
ArrayViews ArrayViews CopyNumScanParams
Examples

```r
require(BSgenome.Hsapiens.UCSC.hg18)
bsgenome <- BSgenome.Hsapiens.UCSC.hg18
require(data.table)
extdir <- system.file("extdata", package="VanillaICE", mustWork=TRUE)
features <- suppressWarnings(fread(file.path(extdir, "SNP_info.csv")))
fr <- GRanges(paste0("chr", features$Chr), IRanges(features$Position, width=1),
isSnps=features["Intensity Only"]==0)
fr <- SnpGRanges(fr)
names(fr) <- features["Name"]
seqlevels(fr) <- setdiff(seqlevels(bsgenome), seqlevels(fr))
seqinfo(fr) <- seqinfo(bsgenome)[seqlevels(fr),]
fr <- sort(fr)
files <- list.files(extdir, full.names=TRUE, recursive=TRUE, pattern="FinalReport")
views <- ArrayViews(rowRanges=fr, sourcePaths=files, parsedPath=tempdir())
show(views)
## read the first file
dat <- fread(files[1])
## information to store on the markers
select <- c("SNP Name", "Allele1 - AB", "Allele2 - AB",
"Log R Ratio", "B Allele Freq")
## which rows to keep in the MAP file. By matching on the sorted GRanges object
## containing the feature annotation, the low-level data for the log R ratios/
## B allele frequencies will also be sorted
##
index_genome <- match(names(fr), dat["SNP Name"])
scan_params <- CopyNumScanParams(index_genome=index_genome, select=select)
## parse the source files
##
parseSourceFile(views, scan_params)
list.files(parsedPath(views))
##
## Inspecting source data through accessors defined on the views object
##
require(oligoClasses)
## log R ratios
r <- head(lrr(views))
## B allele frequencies
b <- head(baf(views))
g <- head(genotypes(views))
```

probability

<table>
<thead>
<tr>
<th>Accessor for probability filter</th>
</tr>
</thead>
</table>

Description

Accessor for probability filter

Usage

```r
probability(object)
```
Arguments

object

a `FilterParam` object

---

Rescale a numeric vector

Description

Rescale a numeric vector

Usage

```
rescale(x, l, u)
```

Arguments

x
numeric vector

l
lower limit of rescaled x

u
upper limit of rescaled x

---

Robust statistics for matrices

Description

Compute the column-wide or row-wise mode of numeric matrices

Compute the median absolute deviation (MAD) for the rows of a matrix

Usage

```
rowModes(x)
```

```
colModes(x)
```

```
rowMAD(x, ...)
```

Arguments

x
matrix

...additional arguments to `rowMedians`

Value

numeric vector

See Also

```
mad
mad rowMedians
```
Examples

X <- matrix(rnorm(100), 10, 10)
rowMAD(X)

segs Accessor for the HMM segments

Description

Accessor to obtain all segments from the HMM.

Usage

segs(object)

Arguments

object see showMethods(segs)

Value

a GRanges-derived object

show, Viterbi-method Show method for objects of class Viterbi

Description

Show method for objects of class Viterbi

Usage

## S4 method for signature 'Viterbi'
show(object)

Arguments

object a Viterbi object
snpArrayAssays

Create an assays object from log R ratios and B allele frequencies

Description

This function is exported primarily for internal use by other BioC packages.

Usage

snpArrayAssays(cn = new("matrix"), baf = new("matrix"), ...)

Arguments

cn
matrix of log R ratios

baf
matrix of B allele frequencies

... additional matrices of the same dimension, such as SNP genotypes.

Examples

data(snp_exp)
r <- lrr(snp_exp)
b <- baf(snp_exp)
sl <- snpArrayAssays(cn=r, baf=b)

SnpArrayExperiment-class

A RangedSummarizedExperiment-derived class of marker-level SNP array data for copy number inference

Description

A RangedSummarizedExperiment-derived class of marker-level SNP array data for copy number inference

Constructor for SnpArrayExperiment

Usage

SnpArrayExperiment(cn, baf, rowRanges = GRanges(), colData = DataFrame(), isSnp = logical(), ...)

## S4 method for signature 'missing'
SnpArrayExperiment(cn, baf, rowRanges = GRanges(),
                    colData = DataFrame(), isSnp = logical(), ...)

## S4 method for signature 'matrix'
SnpArrayExperiment(cn, baf, rowRanges = GRanges(),
                    colData = DataFrame(row.names = colnames(cn)), isSnp = logical(), ...)
Arguments

- `cn`: matrix of copy number estimates (e.g., log R ratios)
- `baf`: matrix of B allele frequencies
- `rowRanges`: GRanges object for SNPs/nonpolymorphic markers
- `colData`: DataFrame containing sample-level covariates
- `isSnp`: logical vector indicating whether marker is a SNP
- `...`: additional arguments passed to `SummarizedExperiment()` constructor function

Examples

```r
## empty container
SnpArrayExperiment()

data(snp_exp) # example
SnpArrayExperiment(cn=lrr(snp_exp), baf=baf(snp_exp),
                   rowRanges=rowRanges(snp_exp))
```

---

**SnpExperiment**

*Constructor for SnpArrayExperiment*

**Description**

A single-argument generic function to construct a SnpArrayExperiment.

**Usage**

```r
SnpExperiment(object)
```

```r
## S4 method for signature 'ArrayViews'
SnpExperiment(object)
```

**Arguments**

- `object`: see `showMethods('SnpExperiment')` for a list of supported objects

**Examples**

```r
view <- ArrayViews()
SnpExperiment(view)
```
**SnpGRanges-class**  

An extension to GRanges for representing SNPs

**Description**

An extension to GRanges for representing SNPs  
Constructor for SnpGRanges class

**Usage**

```
SnpGRanges(object = GRanges(), isSnp, ...)
```

## S4 method for signature 'missing'
```
SnpGRanges(object, isSnp)
```

## S4 method for signature 'GRanges'
```
SnpGRanges(object, isSnp)
```

**Arguments**

- `object`: A GRanges object  
- `isSnp`: A logical vector. Each genomic interval in the GRanges container corresponds to a marker on the genotyping array. `isSnp` is FALSE for nonpolymorphic markers such as those included on the Affymetrix 6.0 chips.  
- `...`: ignored

**Slots**

- `elementMetadata`: a SnpDataFrame

**Examples**

```
SnpGRanges()
g <- GRanges("chr1", IRanges(15L, 15L))  
SnpGRanges(g, isSnp=TRUE)
```

---

**SnpArrayExperiment**  

An example SnpArrayExperiment

---

**Description**

A container for low-level summaries used for downstream copy number estimation, including log R ratios, B allele frequencies, and genotypes

**Format**

- A SnpArrayExperiment object
sourcePaths  

Accessor for file paths containing SNP-level summaries

Description

Files containing SNP-level summaries for log R ratios, B allele frequencies, and genotypes – one sample per subject – are required.

Usage

sourcePaths(object)

Arguments

object an ArrayViews object

Examples

sourcePaths(ArrayViews())

start,oligoSnpSet-method

Retrieve genomic location of SNPs

Description

Retrieve genomic location of SNPs

Usage

## S4 method for signature 'oligoSnpSet'
start(x)

Arguments

x a oligoSnpSet object
state.HmmGRanges-method

Accessor for copy number state

Description

Extract the copy number state for each genomic interval.

Usage

## S4 method for signature 'HmmGRanges'
state(object)

Arguments

object

a HmmGRanges object

state-methods

Accessory for the Viterbi state path

Description

The states are represented as integers: 1=homozygous deletion, 2=hemizygous deletion, 3=diploid normal heterozygosity, 4=diploid region of homozygosity, 5=single copy gain, 6=two or more copy gain.

Usage

## S4 method for signature 'Viterbi'
state(object)

Arguments

object

a Viterbi object

sweepMode

Sweep the modal log R ratio (by row or column) from a matrix of log R ratios

Description

This function simplifies the process of sweeping the modal log R ratio from the rows or columns of a SnpArrayExperiment object. It is most useful when a large number of samples (more than 10) are available and the dataset is a collection of germline samples. We assume that the samples are from a single batch and that the modal value will be a robust estimate of the mean log R ratio for diploid copy number. Variation in the modal estimates between markers is presumed to be attributable to probe effects (e.g., differences hybridization efficiency/PCR do to sequence composition). For sex chromosomes, one should apply this function separately to men and women and then recenter the resulting matrix according to the expected copy number.
Usage
sweepMode(x, MARGIN)

## S4 method for signature 'SnpArrayExperiment'
sweepMode(x, MARGIN)

Arguments
x see showMethods(sweepMode)
MARGIN integer indicating which margin (1=rows, 2=columns) to sweep the mode

Value
an object of the same class as x

Examples
data(snp_exp)
snp_exp_rowcentered <- sweepMode(snp_exp, 1)
snp_exp_colcentered <- sweepMode(snp_exp, 2)
x <- lrr(snp_exp)
x_rowcentered <- sweep(x, 1, rowModes(x))
all.equal(lrr(snp_exp_rowcentered), x_rowcentered)

---

threshold

Threshold numeric values

Description
Threshold numeric values according to user-specific limits. The thresholded values can also be jittered near the limits.

Usage
threshold(x, lim = c(-Inf, Inf), amount = 0)

Arguments
x numeric matrix or vector
lim limit at which to threshold entries in x
amount see jitter

See Also
jitter

Examples
x <- rnorm(1000, 0, 3)
y <- threshold(x, c(-5, 5))
range(y)
TransitionParam

Constructor for TransitionParam class

Description

Contains parameters for computing transition probabilities

Usage

TransitionParam(taup = 1e+10, taumax = 1 - 5e+06)

## S4 method for signature 'TransitionParam'
show(object)

Arguments

taup  
length-one numeric vector

taumax  
The maximum probability that the current state is the same as the preceding
state. See details

object  
a TransitionParam object

Details

Diagonal elements of the transition probability matrix are computed as $e^{-2d/\text{taup}}$, where $d$ is the
distance between markers $i$ and $i-1$ and $\text{taup}$ is typically in the range of $1\times10^6$. This probability
is constrained to be no larger than $\text{taumax}$. The probabilities on the off-diagonal elements are the
same and are subject to the constraint that the rows of the transition probability matrix sum to 1.

Examples

TransitionParam()

## higher values of taup make transitions between states less likely
TransitionParam(taup=1e12)

updateHmmParams

Run the Baum-Welch algorithm to update HMM parameters

Description

This function is not intended to be called directly by the user. It is exported in the package NAMES-
PACE for internal use by other BioC packages.

Usage

updateHmmParams(object, emission_param = EmissionParam(),
transition_param = TransitionParam())
Arguments

- object a SnpArrayExperiment object
- emission_param a EmissionParam object
- transition_param a TransitionParam object

VanillaICE
A hidden markov model for detection of germline copy number variants from arrays

Description
A hidden markov model for detection of germline copy number variants from arrays

viewports
Default viewports for plotting CNV data with lattice-style graphics

Description
Default viewports for plotting CNV data with lattice-style graphics

Usage

viewports()

Value
list

See Also

xyplotList xygrid

Examples

vps <- viewports()
xyplotList  Lattice-style plots for granges and SnpArrayExperiment objects

Description

Data for the graphic is generated by a call to grangesData.

Usage

xyplotList(granges, se, param = HmmTrellisParam())

## S4 method for signature 'HmmGRanges,SnpArrayExperiment'
xyplotList(granges, se,
    param = HmmTrellisParam())

## S4 method for signature 'GRangesList,SnpArrayExperiment'
xyplotList(granges, se,
    param = HmmTrellisParam())

xygrid(trellis_plot, viewports, granges)

Arguments

granges  a HmmGRanges object
se       a SnpArrayExperiment
param    trellis parameters for plotting HMM
trellis_plot  an object of class trellis
viewports  a list of viewports as provided by the viewports function

See Also

viewports

Examples

if(require("BSgenome.Hsapiens.UCSC.hg18")){
  bsgenome <- BSgenome.Hsapiens.UCSC.hg18
  snp_exp <- getExampleSnpExperiment(bsgenome)
  seqlevels(snp_exp, force=TRUE) <- "chr22"
  fit <- hmm2(snp_exp)
  g <- reduce(hemizygous(fit), min.gapwidth=500e3)
  trellis_param <- HmmTrellisParam()
  fig <- xyplotList(g, snp_exp, trellis_param)
  vps <- viewports()
  xygrid(fig[[1]], vps, g)
}
Index

*Topic datasets
  hmmResults, 23
  snp_exp, 35

*Topic manip
  rescale, 31
  '[', ArrayViews, ANY-method
    (ArrayViews-class), 3
  '[', ArrayViews, ANY, ANY, ANY-method
    (ArrayViews-class), 3
  '[', ArrayViews, ANY-method
    (ArrayViews-class), 3
  '$', ArrayViews-method
    (ArrayViews-class), 3
  '$<-', ArrayViews-method
    (ArrayViews-class), 3

  acf, 3
  acf2, 3
  ArrayViews, 12, 29
  ArrayViews (ArrayViews-class), 3
  ArrayViews, numeric, numeric-method
    (ArrayViews-class), 3
  ArrayViews-class, 3

  baf, ArrayViews-method (genotypes), 16
  baf, SnpArrayExperiment-method
    (genotypes), 16
  baf_means (cn_means), 8
  baf_means, ArrayViews-method
    (ArrayViews-class), 3
  baf_means, EmissionParam-method
    (cn_means), 8
  baf_means,HmmParam-method (cn_means), 8
  baf_means<-, numeric-method
    (cn_means), 8
  baf_sds (cn_means), 8
  baf_sds, EmissionParam-method (cn_means), 8
  baf_sds<-, numeric-method
    (cn_means), 8
  bafFile (lrrFile), 27
  bafFile, ArrayViews-method (lrrFile), 27
  baumWelchUpdate, 6
  calculateEmission, 6
  calculateEmission, list-method
    (calculateEmission), 6
  calculateEmission, numeric-method
    (calculateEmission), 6
  calculateEmission, RangedSummarizedExperiment-method
    (calculateEmission), 6
  cn_means, 8
  cn_means, EmissionParam-method
    (cn_means), 8
  cn_means, HmmParam-method (cn_means), 8
  cn_means<-, numeric-method
    (cn_means), 8
  cn_sds (cn_means), 8
  cn_sds, EmissionParam-method (cn_means), 8
  cn_sds, HmmParam-method (cn_means), 8
  cn_sds<-, numeric-method
    (cn_means), 8
  cnvFilter, 7, 16
  cnvFilter, GRanges-method (cnvFilter), 7
  cnvFilter, HMM-method (cnvFilter), 7
  cnvFilter, HmmList-method (cnvFilter), 7
  cnvSegs, 16
  cnvSegs (cnvFilter), 7
  cnvSegs, Hmm-method (cnvFilter), 7
  cnvSegs, HmmGRanges-method (cnvFilter), 7
  cnvSegs, HmmList-method (cnvFilter), 7
  colModes (rowModes), 31
  colnames (ArrayViews-class), 3
  colnames, ArrayViews-method
    (ArrayViews-class), 3
  colnames<-, (ArrayViews-class), 3
  colnames<-, ArrayViews, character-method
    (ArrayViews-class), 3
  copyNumber, SnpArrayExperiment-method
    (genotypes), 16
  CopyNumScanParams, 5, 29
INDEX

CopyNumScanParams
 (CopyNumScanParams-class), 11
CopyNumScanParams-class, 11
deletion (cnvFilter), 7
deletion,HMM-method (cnvFilter), 7
dim, ArrayViews-method
 (ArrayViews-class), 3
doUpdate, 12
dropDuplicatedMapLocs, 13
dropSexChrom, 13
duplication (cnvFilter), 7
duplication,HMM-method (cnvFilter), 7
duplication,HMMList-method (cnvFilter), 7
emission, 14
emission,HmmParam-method (emission), 14
emission<- (emission), 14
emission<-,HMM-method (emission), 14
emission<-,HMMList-method (emission), 7
EmissionParam, J4, 20, 40
EmissionParam (cn_means), 8
emissionParam, HMM-method
 (emissionParam), 14
emissionParam,HmmGRanges-method
 (emissionParam), 14
emissionParam,HmmParam-method
 (emissionParam), 14
EmissionParam, missing-method
 (cn_means), 8
EmissionParam, numeric-method
 (cn_means), 8
emissionParam<- (emissionParam), 14
emissionParam<-,HmmGRanges, EmissionParam-method
 (emissionParam), 14
emissionParam<-,HmmParam,EmissionParam-method
 (emissionParam), 14
EMupdates (cn_means), 8
EMupdates, EmissionParam-method
 (cn_means), 8
EMupdates,HmmParam-method (cn_means), 8
FilterParam, 8
FilterParam (FilterParam-class), 15
FilterParam-class, 15
filters, 16
filters,HMM-method (filters), 16
fread, 11
genotypes, 16
genotypes, ArrayViews-method
 (genotypes), 16
genotypes, SnpArrayExperiment-method
 (genotypes), 16
getExampleSnpExperiment, 17
getHmmparams, 18
getHmmParams,HMM-method (getHmmparams), 18
getHmmParams,HmmParam-method
 (getHmmparams), 18
GRanges, 7
gtFile (llrrFile), 27
gtFile, ArrayViews-method (llrrFile), 27
hemizygous (cnvFilter), 7
hemizygous,HMM-method (cnvFilter), 7
hemizygous,HMMList-method (cnvFilter), 7
HMM, 21
HMM (HMM-class), 18
HMM-class, 18
hmm2, 16, 19, 19, 21
hmm2, ArrayViews-method (hmm2), 19
hmm2, oligoSnpsSet-method (hmm2), 19
hmm2, SnpArrayExperiment-method (hmm2), 19
HMMList, 20, 21
HMMList-class, 21
Hmmparam, 12, 22
Hmmparam, matrix-method (Hmmparam), 22
Hmmparam, missing-method (Hmmparam), 22
hmmResults, 23
Hmmtrellisparam, 23
homzygous (cnvFilter), 7
homzygous,HMM-method (cnvFilter), 7
homzygous,HMMList-method (cnvFilter), 7
IdiogramParams, 23
IdiogramParams-class, 24
isHeterozygous, 25
isHeterozygous, ArrayViews-method
 (isHeterozygous), 25
isHeterozygous, matrix-method
 (isHeterozygous), 25
isHeterozygous, numeric-method
 (isHeterozygous), 25
isHeterozygous, SnpArrayExperiment-method
 (isHeterozygous), 25
jitter, 38
length, LogLik-method (LogLik-class), 26
LogLik, 26, 26, 27
LogLik-class, 26
lrr,ArrayViews-method (genotypes), 16
lrr,SnprArrayExperiment-method (genotypes), 16
lrrFile, 27
lrrFile,ArrayViews-method (lrrFile), 27
lrrFile<- (lrrFile), 27
lrrFile<-,ArrayViews-method (lrrFile), 27
mad, 31
matrixOrNULL, 28
matrixOrNULL-class (matrixOrNULL), 28
NA_filter, 28
NA_filter,character-method (NA_filter), 28
NA_filter,numeric-method (NA_filter), 28
NA_filter,oligoSnpSet-method (NA_filter), 28
NA_filter,SnpArrayExperiment-method (NA_filter), 28
ncol,ArrayViews-method (ArrayViews-class), 3
ncol,HmmParam-method (HmmParam), 22
nrow,ArrayViews-method (ArrayViews-class), 3
nrow,HmmParam-method (HmmParam), 22
numberFeatures,FilterParam-method (numberFeatures), 28
numberFeatures,HMM-method (numberFeatures), 28
numberFeatures,HmmGRanges-method (numberFeatures), 28
parsedPath, 29
parsedPath,ArrayViews-method (parsedPath), 29
parseSourceFile,5,12,29,29
parseSourceFile,ArrayViews,CopyNumScanParams-method (parseSourceFile), 29
plot,IdiogramParams,ANY-method (IdiogramParams), 23
plot,IdiogramParams-method (IdiogramParams), 23
probability, 30
probability,FilterParam-method (FilterParam-class), 15
rescale, 31
rowMAD (row Modes), 31
rowMedians, 31
sapply,ArrayViews-method (ArrayViews-class), 3
segs, 32
segs,HMM-method (segs), 32
segs,HmmGRanges-method (cnvFilter), 7
show,ArrayViews-method (ArrayViews-class), 3
show,CopyNumScanParams-method (CopyNumScanParams-class), 11
show,EmissionParam-method (cn_means), 8
show,FilterParam-method (FilterParam-class), 15
show,HMM-method (HMM-class), 18
show,HmmGRanges-method (HMMList-class), 21
show,HmmParam-method (Hmmparam), 22
show,IdiogramParams-method (IdiogramParams-class), 24
show,LogLik-method (LogLik-class), 26
show,TransitionParam-method (TransitionParam), 39
show,Viterbi-method, 32
snp_exp, 35
snpArrayAssays, 33
SnprArrayExperiment, 17, 20, 28, 40
SnprArrayExperiment (SnprArrayExperiment-class), 33
SnprArrayExperiment,matrix-method (SnprArrayExperiment-class), 33
SnprArrayExperiment,missing-method (SnprArrayExperiment-class), 33
SnprArrayExperiment-class, 33
SnpExperiment, 34
SnpExperiment,ArrayViews-method (SnpExperiment), 34
SnpGRanges (SnpGRanges-class), 35
SnpGRanges,Granges-method (SnpGRanges-class), 35
SnpGRanges,missing-method (SnpGRanges-class), 35
sourcePaths, 36
sourcePaths,ArrayViews-method (sourcePaths), 36
start,ArrayViews-method (ArrayViews-class), 3
start,oligoSnpSet-method, 36
state,FilterParam-method (FilterParam-class), 15
state,HMM-method (HMM-class), 18
state,HmmGRanges-method, 37
state,Viterbi-method (state-methods), 37
INDEX

state-methods, 37
sweepMode, 37
sweepMode, SnpArrayExperiment-method
  (sweepMode), 37

threshold, 38
TransitionParam, 20, 39, 40
TransitionParam, missing-method
  (TransitionParam), 39
TransitionParam, numeric-method
  (TransitionParam), 39

unlist, HMMList-method (HMMList-class),
  21
updateHMMParams, 39

VanillaICE, 40
VanillaICE-package (VanillaICE), 40

viewports, 40, 41

xygrid, 40
xygrid(xyplotList), 41
xyplotList, 40, 41
xyplotList, GRangesList, SnpArrayExperiment-method
  (xyplotList), 41
xyplotList, HmmGRanges, SnpArrayExperiment-method
  (xyplotList), 41