Package ‘VanillaICE’

March 26, 2015

Version 1.29.9
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)
Imports Biobase, oligoClasses (>= 1.24.0), IRanges (>= 1.14.0), S4Vectors, foreach, matrixStats, data.table, grid, lattice, methods, GenomeInfoDb, crlmm
Suggests RUnit, SNPchip, human610quadv1bCrlmm, BSgenome.Hsapiens.UCSC.hg18, ArrayTV
Enhances doMC, doMPI, doSNOW, doParallel, doRedis
License LGPL-2
LazyLoad yes
biocViews CopyNumberVariation
Roxygen list(wrap=FALSE)

## Local Variables
```
## time-stamp-pattern `\`8/Date: %3a %3b %2d %02H:%02M:%02S %Z %:y\{}n`''
## End
NeedsCompilation yes

R topics documented:

ArrayViews-class .................................................. 3
baumWelchUpdate .................................................. 5
calculateEmission .................................................. 6
cnvFilter ........................................................... 6
cn_means ........................................................... 8
constrainMu2 ......................................................... 10
CopyNumScanParams-class .......................................... 11
doUpdate ............................................................ 12
dropDuplicatedMapLocs ............................................. 12
dropSexChrom ........................................................ 13
emission ............................................................. 14
emissionParam ....................................................... 14
fileName ............................................................. 15
FilterParam-class .................................................. 16
filters ................................................................. 17
genotypes ............................................................. 17
generateExampleSnpExperiment ................................... 18
generateHmmParams ................................................ 19
hmm ................................................................. 19
HMM-class ............................................................ 20
hmm2 ................................................................. 21
HmmGRanges ........................................................ 22
HMMList ............................................................. 23
HMMList-class ........................................................ 23
HmmParam ........................................................... 24
HmmResults .......................................................... 25
HmmTrellisParam .................................................... 26
IdiogramParams ...................................................... 27
IdiogramParams-class ............................................... 27
LogLik ............................................................... 28
LogLik-class ........................................................ 29
matrixOrNULL ....................................................... 29
NA_filter ............................................................ 30
numberFeatures ..................................................... 30
parsedPath .......................................................... 31
parseSourceFile ..................................................... 31
probability .......................................................... 32
rescale ............................................................... 33
rowModes ............................................................ 33
segs ................................................................. 34
```

R topics documented:
**ArrayViews-class**

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.

**Usage**

```r
ArrayViews(class = "ArrayViews", colData, rowRanges = GRanges(),
           sourcePaths = character(), scale = 1000, sample_ids,
           parsedPath = getwd())
```

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

colnames(x) <- 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
- **rowRanges**: 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
- **parsedPath**: character vector indicating where parsed files should be saved
- **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
- **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
- **rowData**: A DataFrame. WARNING: The accessor for this slot is rowRanges, not rowData!
- **index**: A GRanges object
baumWelchUpdate

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

See Also
CopyNumScanParams parseSourceFile

Examples

ArrayViews()
## From unit test
require(BSgenome.Hsapiens.UCSC.hg18)
require(data.table)
extrdir <- system.file("extdata", package="VanillaICE", mustWork=TRUE)
features <- suppressWarnings(fread(file.path(extrdir, "SNP_info.csv")))
fgr <- GRanges(paste0("chr", features$Chr), IRanges(features$Position, width=1),
isSnp=features["Intensity Only"]>=0)
fgr <- SnpGRanges(fgr)
names(fgr) <- features["Name"]
seqlevels(fgr) <- seqlevels(BSgenome.Hsapiens.UCSC.hg18)[seqlevels(BSgenome.Hsapiens.UCSC.hg18) %in% seqlevels(fgr)]
seqinfo(fgr) <- seqinfo(BSgenome.Hsapiens.UCSC.hg18)[seqlevels(fgr),]
fgr <- sort(fgr)
files <- list.files(extrdir, full.names=TRUE, recursive=TRUE, pattern="FinalReport")
ids <- gsub("\.*\.*\.*\.*\.*", "", gsub("FinalReport\.*", "", basename(files)))
views <- ArrayViews(rowRanges=fgr,
    sourcePaths=files,
    sample_ids=ids)
## 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  \hspace{1cm} \textit{Calculate the emission probabilities for the 6-state HMM}

\textbf{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.

\textbf{Usage}

```
calculateEmission(x, param = EmissionParam())
```

\textbf{Arguments}

\begin{itemize}
  \item \texttt{x} \hspace{1cm} list of low-level data with two elements: a numeric vector of log R ratios and a numeric vector of B allele frequencies
  \item \texttt{param} \hspace{1cm} parameters for the 6-state HMM
\end{itemize}

\textbf{Value}

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

\textbf{See Also}

baumWelchUpdate

\textbf{cnvFilter  \hspace{1cm} Filter the HMM-derived genomic ranges for copy number variants}

\textbf{Description}

The HMM-derived genomic ranges are represented as a GRanges-derived object. \texttt{cnvFilter} returns a GRanges object using the filters stipulated in the filters argument.

\textbf{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 'HmmlRanges'
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)

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

## S4 method for signature 'HMMList'
cnvFilter(object, filters = FilterParam())

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

Arguments

object see showMethods(cnvFilter)
filters a FilterParam object

See Also

FilterParam

Examples

data(snp_exp)
fit <- hmm2(snp_exp)
segs(fit) ## all intervals
cnvSegs(fit)
filter_param <- FilterParam(probability=0.95, numberFeatures=10, state=c("1", "2"))
cnvSegs(fit, filter_param)
filter_param <- FilterParam(probability=0.5, numberFeatures=2, state=c("1", "2"))
cnvSegs(fit, filter_param)
hemizygous(fit)
homozygous(fit)
duplication(fit)

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)

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)

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

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)
```
Description
Constraints for updating the means of the copy number states
Constraints for updating the standard deviations of the BAFs
Restricted range for CN values

Usage
constrainMu2(mu)
constrainSd2(sigma)
copyNumberLimits(is.log)

Arguments
mu numeric vector of means
sigma numeric vector of standard deviations (non-negative)
is.log whether copy number estimates are on log scale

Value
numeric vector of means
constrained standard deviations
numeric vector giving the lower and upper values of the restricted range

Examples
copyNumberLimits(is.log=TRUE)
copyNumberLimits(is.log=FALSE)
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

```r
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

Slots

- `index_genome`
  - the column label for the log R ratios
- `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  
*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

dropDuplicatedMapLocs(object)
dropSexChrom

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

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)

Description

Removes markers on chromosomes X and Y.

Usage

dropSexChrom(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()
ep_means(ep) <- log2(c(0.1/2, 1/2, 2/2, 2/2, 3/2, 4/2))
emissionParam(hparam) <- ep
```

**fileName**

*Accessor for parsed file(s) containing low level summaries for copy number analysis*

Description

These accessors are not meant to be called directly by the user. They are exported in the package NAMESPACE for internal use by other BioC packages.

Usage

```r
## S4 method for signature 'ArrayViews'
fileName(object, label)

lrrFile(object, label = "lrr")
bafFile(object, label = "baf")
gtFile(object, label = "gt")

## S4 method for signature 'ArrayViews'
lrrFile(object, label = "lrr")

## S4 method for signature 'ArrayViews'
bafFile(object, label = "baf")

## S4 method for signature 'ArrayViews'
gtFile(object, label = "gt")
```

Arguments

- **object**: An array views-type container
- **label**: character string to be incorporated as part of the file name for parsed BAFs

Examples

```r
views <- ArrayViews(parsedPath=tempdir())
sourcePaths(views)
lrrFile(views)
bafFile(views)
gtFile(views)
```
FilterParam-class

Container for the common criteria used to filtering genomic ranges

**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)
```

```r
# S4 method for signature 'FilterParam'
probability(object)
```

```r
# S4 method for signature 'FilterParam'
state(object)
```

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

**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)
## Description
Create an example SnpArrayExperiment from source files containing marker-level genomic data that are provided in this package.

### Usage
```
getExampleSnpExperiment()
```

### Value
A `SnpArrayExperiment`

### Examples
```
## Not run:
snp_exp <- getExampleSnpExperiment()

## 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 <- HHmm()
  getHmmparams(hmm_object)


hmm

Deprecated functions in the VanillaICE package

Description

These functions have been deprecated. The functions are only provided only for compatability with older versions and will be defunct at the next release.

Usage

  hmm(object, ...)

  robustSds(x, takeLog = FALSE, ...)

Arguments

  object         see showMethods(hmm) for a listing of classes for which this method is defined
  ...           additional arguments to mad
  x             a numeric matrix
  takeLog       whether to first log2 transform the numeric matrix

Value

  a matrix of the same dimension as the input. Within a column, the entries are identical

See Also

  mad
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

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

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

```r
## 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

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
HMM()
```
**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 \texttt{hmm2} method allows parallelization across samples using the \texttt{foreach} paradigm. Parallelization is automatic when enabled via packages such as \texttt{snow/doSNOW}.

Examples

\begin{verbatim}
  tp <- TransitionParam()
  TransitionParam(taup=1e12)
  data(snp_exp)
  emission_param <- EmissionParam(temper=1/2)
  fit <- hmm2(snp_exp, emission_param)
  unlist(fit)
  cnvSegs(fit)
  ## 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 \texttt{1} (homozygous deletion) or \texttt{2} (hemizygous deletion)
  fp <- FilterParam(probability=0.5, numberFeatures=2, state=c("1", "2"))
  cnvSegs(fit, fp)
  ## for parallelization
  ## Not run:
  library(snow)
  library(doSNOW)
  c1 <- makeCluster(2, type = "SOCK")
  registerDoSNOW(c1)
  fit <- hmm2(snp_exp, emission_param)
  ## End(Not run)
\end{verbatim}

##HmmGRanges container ##

Description

###HmmGRanges container ###

Usage

\texttt{HmmGRanges(states, feature_starts, feature_chrom, loglik, emission_param = EmissionParam())}

Arguments

\begin{verbatim}
  states   copy number number state inferred by HMM ##
  feature_starts  start location in reference genome [basepairs] ##
  feature_chrom   end location in reference genome [basepairs] ##
\end{verbatim}
loglik the log likelihood ##
emission_param an instance of EmissionParam class ##

Examples

## library(oligoClasses)
## library(IRanges)
## path <- system.file("extdata", package="VanillaICE")
## se <- readRDS(file.path(path, "snp_exp.rds"))
## states <- Rle(factor(c(3, 4, 3, 5, 3, 2, 3, 3, 2, 3, 2, 3)),
##                 as.integer(c(996, 102, 902, 50, 2467, 102, 76, 1822,
##                           99, 900, 20, 160)))
## hgr <- HmmGRanges(states=states, feature_starts=start(se),
##                   feature_chrom=chromosome(se), loglik=15.3)
##

---

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

---

HMMList-class

Class, constructor, and methods for representing HMM results from multiple samples

**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

```r
## 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

```r
data(snps_exp)
fit <- hmm2(snps_exp)
class(fit)
identical(length(fit), ncol(snps_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)
```

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

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

icePlatforms List platforms for which ICE option is supported.

Description
When processing genotypes with the crlmm, confidence scores for the diallelic genotype calls are available. One can estimate the emission probabilities for the crlmm diallelic genotypes using the confidence scores by setting the value of ICE to TRUE in the constructor for the HmmOptionList class. Currently, only certain platforms are supported for this option.

Usage

icePlatforms()

Value
A character vector of the annotation packages that are supported for the ICE option

References
Scharpf, RB et al., 2008, Annals of Applied Statistics

Examples

icePlatforms()
IdiogramParams

Constructor for IdiogramParam objects

Description

Parameters for plotting idiograms

Usage

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

## 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

IdiogramParams-class  Parameter class for plotting idiograms

Description

Parameter class for plotting idiograms

Usage

## S4 method for signature 'IdiogramParams'
show(object)
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
LogLik-class

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

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)
require(data.table)

extdir <- system.file("extdata", package="VanillaICE", mustWork=TRUE)
features <- suppressWarnings fread(file.path(extdir, "SNP_info.csv"))

fgr <- GRanges(paste0("chr", features$Chr), IRanges(features$Position, width=1),
               isSnip=features["Intensity Only"]==0)

fgr <- SnpGRanges(fgr)

names(fgr) <- features["Name"]

seqlevels(fgr) <- seqlevels(BSgenome.Hsapiens.UCSC.hg18)[seqlevels(BSgenome.Hsapiens.UCSC.hg18) %in% seqlevels(fgr),]

seqinfo(fgr) <- seqinfo(BSgenome.Hsapiens.UCSC.hg18)[seqlevels(fgr),]

fgr <- sort(fgr)

files <- list.files(extdir, full.names=TRUE, recursive=TRUE, pattern="FinalReport")
views <- ArrayViews(rowRanges=fgr, sourcePaths=files, parsedPath=tempdir())
show(views)

## read the first file

dat <- fread(files[[1]])

## information to store on the markers
select <- match(c("SNP Name", "Allele1 - AB", "Allele2 - AB", "Log R Ratio", "B Allele Freq"), names(dat))

## 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(fgr), 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**

**Accessor for probability filter**

**Description**

Accessor for probability filter
Usage

probability(object)

Arguments

object a FilterParam object

dataframe 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

dataframe 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
show, Viterbi-method

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

Description

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

Constructor for SnpArrayExperiment

Usage

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

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

## S4 method for signature 'matrix'
SnpArrayExperiment(cn, baf, rowRanges = GRanges(),
colData = DataFrame(row.names = colnames(cn)), isSn = 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
- **isSnps**: logical vector indicating whether marker is a SNP
- **...**: additional arguments passed to initialization method for summarisedExperiment

**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)
```

## S4 method for signature 'ArrayViews'

```r
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)

snp_exp 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**

```r
sourcePaths(object)
```

**Arguments**

- `object` an `ArrayViews` object

**Examples**

```r
sourcePaths(ArrayViews())
```

---

**start, oligoSnpsSet-method**

*Retrieve genomic location of SNPs*

**Description**

Retrieve genomic location of SNPs

**Usage**

```r
## S4 method for signature 'oligoSnpsSet'
start(x)
```

**Arguments**

- `x` an `oligoSnpsSet` object
state,HmmGRanges-method

Accessor for copy number state

Description

Extract the copy number state for each genomic interval.

Usage

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

Arguments

- `object`: a HmmGRanges object

state-methods

Accessor 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

```r
## 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**

```r
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**

```r
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**

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

```r
# 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 $e^{-2*d/taup}$, where $d$ is the distance between markers $i$ and $i-1$ and $taup$ is typically in the range of $1e10$. This probability is constrained to be no larger than $taumax$. 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

```r
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**

```r
viewports()
```

**Value**

list

**See Also**

`xyplotList`, `xygrid`

**Examples**

```r
vps <- viewports()
```

---

**viterbi2Wrapper**

*Wrapper function for fitting the viterbi algorithm*

**Description**

The viterbi algorithm, implemented in C, estimates the optimal state path as well as the forward and backward variables that are used for updating the mean and variances in a copy number HMM. The function `viterbi2Wrapper` should not be called directly by the user. Rather, users should fit the HMM by passing an appropriate container to the method `hmm`. We document the `viterbi2Wrapper` arguments as several of the arguments can be modified from their default value when passed from the `hmm` method through the `...`. In particular, `nupdates`, `p.hom`, and `prOutlierBaf`.

**Usage**

```r
viterbi2Wrapper(index.samples, cnStates, prOutlierBAF = list(initial = 1e-05, 
max = 0.001, maxROH = 1e-05), p.hom = 0.05, is.log, limits, 
normalIndex = 3L, nupdates = 10, tolerance = 5, computeLLR = TRUE, 
returnEmission = FALSE, verbose = FALSE, grFun, matrixFun, snp.index, 
anyNP)
```
Arguments

**index.samples**  
Index for the samples that are to be processed.

**cnStates**  
numeric vector for the initial copy number state means.

**prOutlierBAF**  
A list with elements 'initial', 'max', and 'maxROH' corresponding to the initial estimate of the probability that a B allele frequency (BAF) is an outlier, the maximum value for this parameter over states that do not involve homozygous genotypes, and the maximum value over states that assume homozygous genotypes. This parameter is experimental and could be used to fine tune the HMM for different platforms. For example, the BAFs for the Affy platform are typically more noisy than the BAFs for Illumina. One may want to set small values of these parameters for Illumina (e.g, 1e-5, 1e-3, and 1e-5) and larger values for Affy (e.g., 1e-3, 0.01, 1e-3).

**p.hom**  
numeric: weight for observing homozygous genotypes. For value 0, homozygous genotypes / B allele frequencies have the same emission probability in the 'normal' state as in the states hemizygous deletion and in copy-neutral region of homozygosity. Regions of homozygosity are common in normal genomes. For small values of p.hom, hemizygous deletions will only be called if the copy number estimates show evidence of a decrease from normal.

**is.log**  
logical: Whether the copy number estimates in the r matrix are on the log-scale.

**limits**  
numeric vector of length two specifying the range of the copy number estimates in r. Values of r outside of this range are truncated. See copyNumberLimits.

**normalIndex**  
integer specifying the index for the normal state. Note that states must be ordered by the mean of the copy number state. E.g., state 1 is homozygous deletion (0 copies), state 2 is hemizygous deletion (1 copy), normal (2 copies), ... In a 6-state HMM, normalIndex should be 3.

**nupdates**  
integer specifying the maximum number of iterations for reestimating the mean and variance for each of the copy number states. The number of iterations may be fewer than nupdates if the difference in the log-likelihood between successive iterations is less than tolerance.

**tolerance**  
numeric value for indicating convergence of the log-likelihood. If the difference in the log-likelihood of the observed data given the HMM model at iteration i and i-1 is less than tolerance, no additional updates of model parameters using the EM algorithm is needed.

**computeLLR**  
Logical. Whether to compute a log likelihood ratio (LLR) comparing the predicted state to the normal state. This is calculated post-hoc and is not precisely the likelihood estimated from the Viterbi algorithm. When FALSE, the LLR is not calculated and the algorithm is slightly faster.

**returnEmission**  
Logical. If TRUE, an array of emission probabilities are returned. The dimensions of the array are SNPs, samples, and copy number states.

**verbose**  
Logical. Whether to print some of the details of the processing.

**grFun**  
An R function for coercing the state-path from the HMM to a GRanges object. Takes advantage of lexical scope.

**matrixFun**  
An R function for subsetting the assay data (takes advantage of lexical scope).
xyplotList

snp.index: The SNP indices
anyNP: An indicator for whether any of the markers are nonpolymorphic, and therefore BAFs / genotypes are ignored

Value

A GRanges object if returnEmission is FALSE. Otherwise, an array of emission probabilities is returned.

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

```r
snp_exp <- getExampleSnpExperiment()
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)
```

---

[[,oligoSetList,ANY,ANY-method

*Subset method for deprecated oligoSetList*

---

**Description**

The oligoSetList class is deprecated. Use `SnpArrayExperiment` instead.

**Usage**

```r
## S4 method for signature 'oligoSetList,ANY,ANY'
x[[i, j, ..., exact = TRUE]]
```

**Arguments**

- `x` a oligoSetList object
- `i` A length-one numeric vector specifying which chromosome to extract
- `j` A numeric vector specifying samples to extract (optional)
- `...` Ignored
- `exact` Ignored
Index

*Topic datasets
  hmmResults, 25
  snp_exp, 37
*Topic manip
  constrainMu2, 10
  rescale, 33
*Topic misc
  icePlatforms, 26
*Topic smooth
  viterbi2Wrapper, 43
  ‘[’, ArrayViews, ANY-method
  (ArrayViews-class), 3
  [, ArrayViews, ANY, ANY, ANY-method
  (ArrayViews-class), 3
  [, ArrayViews, ANY-method
  (ArrayViews-class), 3
  [[], oligoSetList, ANY, ANY-method, 46

ArrayViews, 12, 31
ArrayViews (ArrayViews-class), 3
ArrayViews, numeric, numeric-method
  (ArrayViews-class), 3
ArrayViews-class, 3

baf, ArrayViews-method (genotypes), 17
baf, SnpArrayExperiment-method
  (genotypes), 17
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<- (cn_means), 8
baf_means<-, EmissionParam, numeric-method
  (cn_means), 8
baf_sds (cn_means), 8
baf_sds, EmissionParam-method
  (cn_means), 8
baf_sds, HmmParam-method (cn_means), 8

baf_sds<-(cn_means), 8
baf_sds<-, EmissionParam, numeric-method
  (cn_means), 8
baffile (fileName), 15
baffile, ArrayViews-method (fileName), 15
baumWelchUpdate, 5

calculateEmission, 6
calculateEmission, list-method
  (calculateEmission), 6
calculateEmission, numeric-method
  (calculateEmission), 6
calculateEmission, SummarizedExperiment-method
  (calculateEmission), 6
cn_means, 8
cn_means, EmissionParam-method
  (cn_means), 8
cn_means, HmmParam-method (cn_means), 8
cn_means<-(cn_means), 8
cn_means<-, EmissionParam, numeric-method
  (cn_means), 8
cn_sds (cn_means), 8
cn_sds, EmissionParam-method (cn_means), 8
cn_sds<-(cn_means), 8
cn_sds<-, EmissionParam, numeric-method
  (cn_means), 8
cnvFilter, 6, 17
cnvFilter, GRanges-method (cnvFilter), 6
cnvFilter, HMM-method (cnvFilter), 6
cnvFilter, HMMList-method (cnvFilter), 6
cnvSegs, 17
cnvSegs (cnvFilter), 6
cnvSegs, HMM-method (cnvFilter), 6
cnvSegs, HMMGRanges-method (cnvFilter), 6
cnvSegs, HMMList-method (cnvFilter), 6
colModes (rowModes), 33
colnames<-(ArrayViews-class), 3
colnames<-.ArrayViews, character-method
   (ArrayViews-class), 3
constrainMu2, 10
constrainSd2 (constrainMu2), 10
copyNumber,SnpArrayExperiment-method
   (genotypes), 17
copyNumberLimits (constrainMu2), 10
CopyNumScanParams, 5, 31
CopyNumScanParams
   (CopyNumScanParams-class), 11
CopyNumScanParams-class, 11
deletion (cnvFilter), 6
deletion, HMM-method (cnvFilter), 6
dim, ArrayViews-method
   (ArrayViews-class), 3
doUpdate, 12
dropDuplicatedMapLocs, 12
dropSexChrom, 13
duplication (cnvFilter), 6
duplication, HMM-method (cnvFilter), 6
duplication, HMMList-method (cnvFilter), 6
emission, 14
emission, HMMParam-method (emission), 14
emission<- (emission), 14
emission<-, HMM-method (emission), 14
emission<-, HMMParam-method (emission), 14
EmissionParam, 14, 21, 42
EmissionParam (cn_means), 8
emissionParam, 14
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
EMupdates, EmissionParam-method
   (cn_means), 8
EMupdates, HMMParam-method (cn_means), 8
fileName, 15
fileName, ArrayViews-method (fileName), 15
FilterParam, 7
FilterParam (FilterParam-class), 16
FilterParam-class, 16
filters, 17
filters, HMM-method (filters), 17
fread, 11
genotypes, 17
genotypes, ArrayViews-method
   (genotypes), 17
genotypes, SnpArrayExperiment-method
   (genotypes), 17
getExampleSnpExperiment, 18
getHmmParams, 19
getHmmParams, HMM-method (getHmmParams), 19
getHmmParams, HMMParam-method
   (getHmmParams), 19
GRanges, 16
gtFile (fileName), 15
gtFile, ArrayViews-method (fileName), 15
hemizygous (cnvFilter), 6
hemizygous, HMM-method (cnvFilter), 6
hemizygous, HMMList-method (cnvFilter), 6
HMM, 23, 24
HMM (HMM-class), 20
hmm, 19
hmm, BaflrrSetList-method (hmm), 19
hmm, BeadStudioSet-method (hmm), 19
hmm, BeadStudioSetList-method (hmm), 19
hmm, oligoSetList-method (hmm), 19
hmm, oligoSnpSet-method (hmm), 19
hmm, SnpSet2-method (hmm), 19
HMM-class, 20
hmm2, 17, 20, 21, 23
hmm2, ArrayViews-method (hmm2), 21
hmm2, oligoSnpSet-method (hmm2), 21
hmm2, SnpArrayExperiment-method (hmm2), 21
hmm2, EmissionParam-method (hmm2), 14
HMMRanges, 22
INDEX

HMMList, 23, 23
HMMList-class, 23
Hmmparam, 12, 24
Hmmparam, matrix-method (Hmmparam), 24
Hmmparam, missing-method (Hmmparam), 24
hmmsResults, 25
HmmtrellisParam, 26
homozygous (cnvFilter), 6
homozygous, HMM-method (cnvFilter), 6
homozygous, HMMList-method (cnvFilter), 6
icePlatforms, 26
IdiogramParams, 27
IdiogramParams-class, 27
jitter, 47
length, LogLik-method (LogLik-class), 29
LogLik, 28, 29
LogLik-class, 29
lrr, ArrayViews-method (genotypes), 17
lrr, SnpArrayExperiment-method (genotypes), 17
lrrFile (fileName), 15
lrrFile, ArrayViews-method (fileName), 15
mad, 19, 34
matrixOrNULL, 29
matrixOrNULL-class (matrixOrNULL), 29
NA_filter, 30
NA_filter, character-method (NA_filter), 30
NA_filter, list-method (NA_filter), 30
NA_filter, numeric-method (NA_filter), 30
NA_filter, oligoSnpSet-method (NA_filter), 30
NA_filter, SnpArrayExperiment-method (NA_filter), 30
ncol, ArrayViews-method (ArrayViews-class), 3
ncol, Hmmparam-method (Hmmparam), 24
nrow, ArrayViews-method (ArrayViews-class), 3
nrow, Hmmparam-method (Hmmparam), 24
numberFeatures, 30
numberFeatures, FilterParam-method (numberFeatures), 30
numberFeatures, HMM-method (numberFeatures), 30
numberFeatures, HmmGRanges-method (numberFeatures), 30
parsedPath, 31
parsedPath, ArrayViews-method (parsedPath), 31
parseSourceFile, 5, 12, 31
parseSourceFile, ArrayViews, CopyNumScanParams-method (parseSourceFile), 31
plot, IdiogramParams, ANY-method (IdiogramParams), 27
plot, IdiogramParams-method (IdiogramParams), 27
probability, 32
probability, FilterParam-method (FilterParam-class), 16
rescale, 33
robustSds (hmm), 19
rowMAD (rowModes), 33
rowMedians, 34
rowModes, 33
sapply, ArrayViews-method (ArrayViews-class), 3
segs, 34
segs, HMM-method (segs), 34
segs, HMMList-method (cnvFilter), 6
show, ArrayViews-method (ArrayViews-class), 3
show, CopyNumScanParams-method (CopyNumScanParams-class), 11
show, EmissionParam-method (cn_means), 8
show, FilterParam-method (FilterParam-class), 16
show, HMM-method (HMM-class), 20
show, HMMList-method (HMMList-class), 23
show, Hmmparam-method (Hmmparam), 24
show, IdiogramParams-method (IdiogramParams-class), 27
show, LogLik-method (LogLik-class), 29
show, TransitionParam-method (TransitionParam), 41
show, Viterbi-method, 34
snp_exp, 37
snpArrayAssays, 35
SnpArrayExperiment, 18, 21, 30, 42, 46
SnpArrayExperiment (SnpArrayExperiment-class), 35
SnpArrayExperiment, matrix-method
(SnpArrayExperiment-class), 35
SnpArrayExperiment, missing-method
(SnpArrayExperiment-class), 35
SnpArrayExperiment-class, 35
SnpExperiment, 36
SnpExperiment, ArrayViews-method
(SnpExperiment), 36
SnpGRanges (SnpGRanges-class), 37
SnpGRanges, GRanges-method
(SnpGRanges-class), 37
SnpGRanges, missing-method
(SnpGRanges-class), 37
SnpGRanges-class, 37
sourcePaths, 38
sourcePaths, ArrayViews-method
(sourcePaths), 38
start, ArrayViews-method
(ArrayViews-class), 3
start, oligoSnpsSet-method, 38
state, FilterParam-method
(FilterParam-class), 16
state, HMM-method (HMM-class), 20
state, HmmGRanges-method, 39
state, Viterbi-method (state-methods), 39
state-methods, 39
sweepMode, 40
sweepMode, SnpArrayExperiment-method
(sweepMode), 40
threshold, 41
TransitionParam, 21, 41, 42
TransitionParam, missing-method
(TransitionParam), 41
TransitionParam, numeric-method
(TransitionParam), 41
unlist, HMMList-method (HMMList-class), 23
updateHmmParams, 42
VanillaICE, 42
VanillaICE-package (VanillaICE), 42
viewports, 43, 45
viterbi2Wrapper, 43
xygrid, 43
xygrid(xyplotList), 45
xyplotList, 43, 45