

VanillaICE

April 19, 2009

HmmOptions-class *Class "HmmOptions"*

Description

Options for fitting the hidden Markov model.

Objects from the Class

Objects can be created by calls of the form `new("HmmOptions", snpset, states, copyNumber.location, copyNumber.scale, copyNumber.ICE, calls.ICE, probHomCall, term5)`.

Slots

states: Object of class "character". Hidden states of the HMM.

snpset: "SnpLevelSet" instance.

copyNumber.location: Object of class "numeric". Ignored when snpset is an instance of SnpCallSet. See [copyNumber.location](#)

copyNumber.scale: Object of class "NULLOrNumeric". This slot is not currently used. If one has standard errors for the copy number estimates, the inverse of the standard errors should be stored in the `cnConfidence` `assayData` element of the snpset object.

copyNumber.ICE: Object of class "logical". See [copyNumber.ICE](#).

calls.ICE: Object of class "logical". See [calls.ICE](#)

probHomCall: Object of class "numeric". The probability of a homozygous genotype call corresponding to each of the states. When `calls.ICE` is TRUE, only the first two elements will be used.

term5: Object of class "NULLOrNumeric". User-specified probabilities of $f(\text{HOM} \mid \text{genotype call, state})$. Ignored when `calls.ICE` is FALSE. For more details, see Equation 2.7 in the manuscript that this package references.

Methods

```

[ signature(x = "HmmOptions"):
annotation signature(object = "HmmOptions"):
annotation<- signature(object = "HmmOptions", value = "ANY"):
calculateDistance signature(object = "HmmOptions"):
calls.ICE signature(object = "HmmOptions"):
calls.ICE<- signature(object = "HmmOptions", value="logical"):
copyNumber.ICE signature(object = "HmmOptions"):
copyNumber.ICE<- signature(object = "HmmOptions", value="logical"):
db signature(object = "HmmOptions"):
hmm signature(object = "HmmOptions", params = "HmmParameter"):
initialize signature(.Object = "HmmOptions"):
notes signature(object = "HmmOptions"):
show signature(object = "HmmOptions"):
states signature(object = "HmmOptions"):
states<- signature(object = "HmmOptions", value = "character"):

```

Author(s)

R. Scharpf

See Also

[HmmParameter-class](#)

Examples

```
showClass("HmmOptions")
```

HmmOptions-methods *Accessors and replacement methods for HmmOptions*

Description

Accessors and replacement methods for HmmOptions objects

Usage

```

copyNumber.location(object)
copyNumber.location(object) <- value
probHomCall(object)
probHomCall(object) <- value
snpset(object)
snpset(object) <- value
term5(object)
term5(object) <- value

```

Arguments

object HmmpOptions instance
 value see the signature of the corresponding methods

Value

copyNumber.location, probHomCall and term5 each return numeric vectors.
 snpset returns an instance of SnpLevelSet

See Also

[HmmpOptions-class](#)

HmmpParameter-class *Class "HmmpParameter"*

Description

Parameters needed for the Viterbi algorithm, including emission probabilities, the genomic distance, and the initial state probabilities

Objects from the Class

Objects can be created by calls of the form `new("HmmpParameter", states, initialStateProbability, emission, genomicDistance, transitionScale)`.

Slots

states: Object of class "character"
initialStateProbability: Object of class "numeric"
emission: Object of class "array"
genomicDistance: Object of class "numeric"
transitionScale: Object of class "matrix"

Methods

```
[ signature(x = "HmmpParameter"):
emission signature(object = "HmmpParameter"):
emission<- signature(object = "HmmpParameter", value = "array"):
genomicDistance signature(object = "HmmpParameter"):
genomicDistance<- signature(object = "HmmpParameter", value = "numeric"):
hmm signature(object = "HmmpOptions", params = "HmmpParameter"):
initialize signature(.Object = "HmmpParameter"):
pi signature(object = "HmmpParameter"):
show signature(object = "HmmpParameter"):
states signature(object = "HmmpParameter"):
states<- signature(object = "HmmpParameter", value = "character"):
tau signature(object = "HmmpParameter"):
transitionScale signature(object = "HmmpParameter"):
transitionScale<- signature(object = "HmmpParameter", value = "matrix"):
```

Note

See Hmmpredict vignette

Author(s)

RS

See Also

[Hmmpredict-class](#)

Examples

```
showClass("Hmmpredict")
```

Hmmpredict-class *Class for the hidden Markov model results*

Description

Contains the predictions for the hidden states and a list of breakpoint matrices. Each element in the list corresponds to the breakpoints identified in one sample.

Objects from the Class

Objects can be created by calls of the form `new("Hmmpredict", states, predictions, breakpoints, SnpClass)`.

Slots

states: Object of class character. Character string of the hidden states.

predictions: Object of class matrix. Matrix of predictions.

breakpoints: Object of class list. List of matrices. Each element is the matrix of breakpoints for one sample.

SnpClass: Object of class character. The class of SNP data (e.g., oligoSnpSet)

featureData: Object of class AnnotatedDataFrame

Methods

SnpClass signature(object = "Hmmpredict"): Accessor for class of SNP data

[signature(x = "Hmmpredict"): Subsetting the predictions

breakpoints signature(object = "Hmmpredict"): Accessor for the list of breakpoints

breakpoints<- signature(object = "Hmmpredict", value = "data.frame"): Replacement method for breakpoints

fData signature(object="Hmmpredict"): Accessor for the data frame of all feature-level (SNP-level) annotation (e.g., physical position)

featureData signature(object = "Hmmpredict"): Accessor for the featureData

featureNames signature(object = "Hmmpredict"): Character string of SNP identifiers

getPar signature(object = "HmmPredict"): Graphical parameters.
initialize signature(.Object = "HmmPredict")
position signature(object = "HmmPredict"): Accessor for the physical position of SNPs in the object HmmPredict
predictions signature(object = "HmmPredict"): Accessor for the predictions
predictions<- signature(object = "HmmPredict", value="matrix"): Replacement method for predictions
sampleNames signature(object = "HmmPredict"): Sample names
show signature(object = "HmmPredict")
states signature(object = "HmmPredict"): Accessor for hidden states

Author(s)

R. Scharpf

See Also

[hmm](#)

Examples

```
showClass("HmmPredict")
```

addFeatureData *Adds chromosomal arm annotation to the featureData*

Description

Adds chromosomal arm annotation to the featureData

Usage

```
addFeatureData(snpset)
```

Arguments

snpset An object inherited from class SnpLevelSet

Value

An AnnotatedDataFrame

Author(s)

R. Scharpf

See Also

[class.AnnotatedDataFrame](#)

calculateBreakpoints

Calculate breakpoints for altered states from the prediction matrix

Description

Wrapper for findBreaks

Usage

```
calculateBreakpoints(object, ...)
```

Arguments

object	Object of class HmmPredict
...	Not implemented

Details

Returns a matrix with information on size of region (Mb), predicted state, number of SNPs in region, chromosome number, and position.

Value

A matrix

Author(s)

R. Scharpf

See Also

[findBreaks](#)

calculateCnSE

Calculates copy number standard errors

Description

Calculates SNP-specific standard errors for copy number using a reference set.

Usage

```
calculateCnSE(object, referenceSet, epsilon = 0.1)
```

Arguments

object	An object of class <code>SnpcopyNumberSet</code> or <code>oligoSnpcopySet</code> .
referenceSet	An object of class <code>SnpcopyNumberSet</code> or <code>oligoSnpcopySet</code> . If missing, this function uses the samples in the <code>object</code> to estimate across sample standard deviations
epsilon	minimum standard error

Details

Calculates SNP-specific standard errors from a reference distribution as the product of the within copy number standard deviation of the sample (test set) and the across samples standard deviation from a reference set. The across samples standard deviation is scaled by the median (e.g., across sample sd = across sample sd / median(across sample sd)).

Value

A matrix of standard errors

Author(s)

R. Scharpf

`calculateDistance` *Calculates the physical distance between adjacent SNPs*

Description

Calculates the distance between adjacent SNPs.

Usage

```
calculateDistance(object)
physicalDistance(object)
```

Arguments

object	Object inheriting from <code>SnpcopyLevelSet</code>
--------	---

Details

Calculates the physical distance between adjacent SNPs along a chromosome.

Value

A vector of integers

Note

The transition probabilities in the Viterbi algorithm are calculated as a function of the physical distance (d) between adjacent SNPs.

probability of staying in state S = $\exp(-2 * d/100 * 1e6)$

Author(s)

R. Scharpf

See Also

[viterbi](#)

calls.ICE

Indicates whether to use confidence scores of the observed copy number estimates in the HMM

Description

In the current framework, the confidence scores are assumed to be the inverse of the standard error

Usage

```
calls.ICE(object)
calls.ICE(object) <- value
```

Arguments

object	An object of class <code>HmmOptions</code>
value	Logical. If TRUE, confidence scores of the genotype calls are used to calculate emission probabilities

Value

Logical

Author(s)

R. Scharpf

See Also

See the `VanillaICE` vignette

calls.emission	<i>Calculates emission probabilities for the genotype calls</i>
----------------	---

Description

Emission probabilities for the genotype calls

Usage

```
calls.emission(object)
```

Arguments

object Object inheriting from class `HmmOptions`

Details

If `calls.ICE = TRUE` in the `HmmOptions` object, then the emission probabilities for the genotype calls take into account the confidence scores for the called genotypes.

Value

Array of emission probabilities for the genotype calls (on the log scale)

Author(s)

R. Scharpf

See Also

[calls.ICE](#)

chromosome1	<i>Simulated chromosome 1</i>
-------------	-------------------------------

Description

Simulated genotype calls and copy number estimates for chromosome 1 (Affymetrix 100k platform).

Usage

```
data(chromosome1)
```

Details

The simulation comprises one subject's genotype, copy number, and confidence scores for 9165 SNPs on chromosome 1. A description of the 5 features simulated in chromosome 1, referred to by regions A-E, and the underlying hidden states in these regions follows.

Genotype calls: With the exception of three regions in this chromosome, we simulated 9165 genotypes (the approximate number of SNPs in the two 50k SNP chips) from a Bernoulli distribution with probability 0.7 of a homozygous genotype. Confidence scores for genotype estimates were obtained by random draws of confidence scores in the Hapmap data when the CRLMM genotype calls agreed with the gold-standard as defined by consensus of the HapMap genotyping centers.

Copy number: The Affymetrix CNAT tool (version 3.0) was used to obtain copy number estimates for the 9165 SNPs from a presumably normal individual in the HapMap dataset (sample identifier NA06993). Deletions and amplifications were simulated from Gaussian distributions with location parameters $\log_2(1)$ and $\log_2(3)$, respectively. For the scale parameter, we used a robust estimate of the \log_2 transformed copy number standard deviation. To illustrate how a confidence score such as a standard error of the copy number estimate could be useful, we simulated standard errors from a shifted Gamma: $\text{Gamma}(1, 2) + 0.3$, where 1 is the shape parameter and 2 is the rate parameter. To ascertain the effect of qualitatively high confidence scores on the ICE HMM, we scaled the robust standard deviation estimate by 1/2. Similarly, to simulate less precise copy number estimates, we multiplied the robust standard deviation estimate by a factor of 2.

For additional information on the five abnormalities simulated in this chromosome, see the manuscript referenced below.

References

RB Scharpf et al. (2008) Hidden Markov Models for the assessment of chromosomal alterations using high-throughput SNP arrays, *Annals of Applied Statistics*

Examples

```
data(chromosome1)
chromosome1
```

copyNumber.ICE	<i>Indicates whether to use confidence scores of the observed copy number estimates in the HMM</i>
----------------	--

Description

In the current framework, the confidence scores are assumed to be the inverse of the standard error

Usage

```
copyNumber.ICE(object)
copyNumber.ICE(object) <- value
```

Arguments

object	An object of class <code>HmmOptions</code>
value	Logical. If TRUE, confidence scores of the copy number estimates are used to calculate emission probabilities.

Details

If TRUE, the confidence scores (assumed to be inverse standard errors) are used by the hidden Markov model. This has the desirable consequence of providing more local smoothing for precise estimates of copy number and more global smoothing for copy number estimates with higher standard errors. When FALSE, a robust estimate of the chip variability is calculated across all of the autosomes.

Value

Logical

Author(s)

R. Scharpf

See Also

See the VanillaICE vignette

`copyNumber.emission`

Calculate emission probabilities for copy number

Description

Calculate emission probabilities for the copy number

Usage

```
copyNumber.emission(object)
```

Arguments

`object` Instance of class `HmmOptions`

Details

If `copyNumber.ICE = TRUE` in the `HmmOptions` object, then the emission probabilities for the copy number estimates take into account the confidence scores stored in the `cnConfidence` slot of the object inheriting from class `SnplLevelSet`. The confidence scores are assumed to be inverse standard errors.

Value

Array of emission probabilities for the copy number estimates (on the log scale)

Author(s)

R. Scharpf

See Also

[copyNumber.ICE](#)

copynumberEmission *Emission probabilities for copy number*

Description

Emission probabilities for copy number

Usage

```
copynumberEmission(copynumber, states, mu, uncertainty, takeLog, verbose = TRUE)
```

Arguments

copynumber	class matrix
states	class character
mu	class numeric: mean of hidden states for Gaussian
uncertainty	uncertainty of copy number estimates
takeLog	whether to take the log of the copy number AND the mu argument
verbose	logical

Value

array	Array of emission probabilities. Dimension 1: SNPs, Dimension 2: samples, Dimension3: states
-------	--

emission *Accessor for emission probabilities*

Description

Accessor for emission probabilities for the hidden Markov model

Usage

```
emission(object)
emission(object) <- value
```

Arguments

object	An object of class <code>HmmParameter</code>
value	An array of emission probabilities (log scale)

Value

An array.

Author(s)

R. Scharpf

See Also

[HmmParameter-class](#), [HmmOptions-class](#)

findBreaks	<i>Identify breakpoints from the hidden Markov model predictions</i>
------------	--

Description

Identify breakpoints: physical position of breaks, number of SNPs in region, and the called hidden state.

Usage

```
findBreaks(x, states, position, chromosome, sample, lik1, lik2)
```

Arguments

x	matrix of predicted hidden states (integer)
states	character string giving the hidden states
position	physical position of the SNPs
chromosome	chromosome number
sample	sample name
lik1	likelihood under alternative model
lik2	likelihood under NULL model

Value

data.frame	data.frame describing regions of normal and altered states
------------	--

fit	<i>Object of class HmmPredict</i>
-----	-----------------------------------

Description

Object of class `HmmPredict`

Usage

```
data(fit)
```

Details

This object was created from the `chromosome1` as described in the `VanillaICE` vignette.

See Also

[chromosome1](#)

Examples

```
data(fit)
```

genomicDistance *A rough estimate of the genomic distance between SNPs*

Description

Calculates the genomic distance between SNPs as a function of the physical distance. This distance is used for calculating transition probabilities in the viterbi algorithm.

Usage

```
genomicDistance(object)
tau(object)
genomicDistance(object) <- value
```

Arguments

object An object inheriting from `HmmOptions`
value A vector of length $T - 1$, where T is the number of SNPs

Details

The genomic distance is calculated as $\exp(-2*d)$, where d is the physical distance in 100MB units. In the Viterbi algorithm, the probability of remaining in state S at SNP $t+1$ is the genomic distance. The probability of transitioning to one of the alternative states is $1 -$ the genomic distance. The `transitionScale` function can be used to give more weight to transitioning back to the baseline ('normal') state.

Value

A numeric vector with length $T-1$.

Author(s)

R. Scharpf

See Also

[transitionScale](#)

genotypeEmission *Emission probabilities for di-allelic genotype calls*

Description

Emission probabilities for di-allelic genotype calls

Usage

```
genotypeEmission(genotypes, states, probHomCall, probMissing, verbose=TRUE)
```

Arguments

genotypes	matrix of integers (1=AA, 2=AB, 3=BB, 4=other)
states	character string of hidden states
probHomCall	numeric: probability of a homozygous genotype call specified in the same order as the hidden states
probMissing	numeric: probability of a missing genotype call specified in the same order as the hidden states
verbose	logical

Details

By default, missing genotype calls will be assumed to be independent of the hidden state. This is not always appropriate, but is easiest to implement in terms of a default.

Value

array	Array of emission probabilities. Dimension 1: SNPs, Dimension 2: samples, Dimension3: states
-------	--

getChromosomeArm *Get the chromosome arm*

Description

Retrieves the chromosome arm (p or q) for each SNP.

Usage

```
getChromosomeArm(snpset)
```

Arguments

snpset	An object inheriting from class SnpLevelSet
--------	---

Value

A vector of class 'character'.

Note

This function uses data(chromosomeAnnotation) in the SNPchip package to retrieve this information.

Author(s)

RS

See Also

[chromosomeAnnotation](#)

hmm

Fits the hidden Markov model

Description

Fits the hidden Markov model

Usage

```
hmm(object, params)
```

Arguments

object	An object of class <code>HmmOptions</code>
params	An object of class <code>HmmParameter</code>

Details

None yet.

Value

An object of class `HmmPredict`

Author(s)

R. Scharpf

References

RB Scharpf et al. (2008) Hidden Markov Models for the assessment of chromosomal alterations using high-throughput SNP arrays, *Annals of Applied Statistics*

See Also

[HmmParameter-class](#) [HmmPredict-class](#) [SnpLevelSet-class](#)

Examples

```
data(chromosome1)
chromosome1 <- chromosome1[1:500, ]
options <- new("HmmOptions",
              states=c("D", "N", "L", "A"),
              snpset=chromosome1,
              copyNumber.location=c(1, 2, 2, 3),
              probHomCall=c(0.99, 0.7, 0.99, 0.7))
validObject(options)
params <- new("HmmParameter",
             states=states(options),
             initialStateProbability=0.99)
cn.emission <- copyNumber.emission(options)
gt.emission <- calls.emission(options)
emission(params) <- cn.emission + gt.emission ##log scale
```



```
genomicDistance(params) <- exp(-2 *calculateDistance(options)/(100*1e6))
##no scaling
transitionScale(params) <- matrix(1, length(states(options)), length(states(options)))
if(validObject(params)) fit <- hmm(options, params)
```

Pi

Accessor for the initial state probabilities of the hidden Markov model

Description

Accessor for the initial state probabilities of the hidden Markov model

Usage

```
Pi(object)
```

Arguments

object object of class `HmmParameter`

Value

Numeric vector with length equal to the number of hidden states

Author(s)

RS

See Also

[HmmParameter-class](#)

scaleTransitionProbability

Scales the probability of transitioning between two states in the Viterbi algorithm

Description

Function only returns a matrix of 1's (by default, no scaling of the transition probabilities). This may be changed in future releases.

Usage

```
scaleTransitionProbability(states, SCALE = 2, normalLabel = "N")
```

Arguments

states character string of hidden states
SCALE Numeric. See details.
normalLabel Character: label used for the baseline state.

Value

A matrix of dimension $S \times S$, where S is the number of states. The diagonal should always be 1 (we do not scale the probability of remaining in the same state). A slot for storing this matrix is provided in the `HmmParameter` class.

Author(s)

RS

See Also

[genomicDistance](#), [transitionScale](#), [HmmParameter-class](#)

states	<i>Accessor and replacement method for the hidden states of the Markov model</i>
--------	--

Description

The latent states for the hidden Markov model.

Usage

```
states(object)
states(object) <- value
```

Arguments

object	An object of class <code>HmmOptions</code> or <code>HmmParameter</code>
value	Vector of class <code>character</code> .

Value

Character string of hidden states

Author(s)

RS

transitionScale	<i>Scales the transition probabilities of the hidden Markov model</i>
-----------------	---

Description

Accessor and replacement methods for scaling the transition probabilities

Usage

```
transitionScale(object)
transitionScale(object) <- value
```

Arguments

object	An object of class <code>HmmParameter</code>
value	A matrix of dimension STATES x STATES, where STATES is the number of hidden states.

Details

The probability of remaining in the same state, $P(S_t = S_{t+1})$ (the diagonal of the transition probability matrix) is a function of the distance between SNPs. The probability of transitioning to some other state is epsilon, where $\text{epsilon} = 1 - P(S_t = S_{t+1})$. The epsilon is split among STATES-1 states. By default, the probability of transitioning from an altered state back to the normal state is twice as likely as the probability of transitioning between two altered states. The weights for epsilon are provided in the `transitionScale` matrix in objects of class `HmmParameter`.

Value

A matrix

Author(s)

R. Scharpf

See Also

[scaleTransitionProbability](#)

viterbi	<i>viterbi algorithm</i>
---------	--------------------------

Description

The Viterbi algorithm for computing the most likely state sequence given a model

Usage

```
viterbi(initialStateProbs, emission, tau, arm, tau.scale, verbose = TRUE, return
```

Arguments

<code>initialStateProbs</code>	initial state probabilities (log scale)
<code>emission</code>	matrix of log emission probabilities (one sample is a matrix)
<code>tau</code>	transition probabilities (original scale)
<code>arm</code>	numeric or character string indicating chromosomal arm
<code>tau.scale</code>	matrix to scale the probability of transitioning between states.
<code>verbose</code>	Logical
<code>returnLikelihood</code>	whether to return the 'loglikelihood'

Details

The Viterbi algorithm is fit independently to each chromosomal arm if `arm` is specified.

Argument `tau.scale` is a matrix that scales the probability of transitioning from an altered state to a normal state to the probability of transitioning between two altered states. If missing, `tau.scale` is 1 (no scaling)

`returnLikelihood` is experimental

Value

`matrix` predicted states

Author(s)

R. Scharpf

Index

*Topic classes

HmmOptions-class, 1
HmmParameter-class, 3
HmmPredict-class, 4

*Topic datasets

chromosome1, 9
fit, 13

*Topic manip

addFeatureData, 5
calculateBreakpoints, 6
calculateCnSE, 6
calculateDistance, 7
findBreaks, 13
getChromosomeArm, 15
scaleTransitionProbability,
17

*Topic methods

calculateBreakpoints, 6
calculateDistance, 7
calls.emission, 9
calls.ICE, 8
copyNumber.emission, 11
copyNumber.ICE, 10
copynumberEmission, 12
emission, 12
genomicDistance, 14
genotypeEmission, 14
HmmOptions-methods, 2
Pi, 17
states, 18
transitionScale, 19

*Topic models

hmm, 16
viterbi, 19

[, HmmOptions-method
(*HmmOptions-class*), 1
[, HmmParameter-method
(*HmmParameter-class*), 3
[, HmmPredict-method
(*HmmPredict-class*), 4

addFeatureData, 5
annotation, HmmOptions-method
(*HmmOptions-class*), 1

annotation<- , HmmOptions, ANY-method
(*HmmOptions-class*), 1

breakpoints (*HmmPredict-class*), 4
breakpoints, HmmPredict-method
(*HmmPredict-class*), 4
breakpoints<- (*HmmPredict-class*),
4

breakpoints<- , HmmPredict, ANY-method
(*HmmPredict-class*), 4
breakpoints<- , HmmPredict-method
(*HmmPredict-class*), 4

calculateBreakpoints, 6
calculateBreakpoints, HmmPredict-method
(*HmmPredict-class*), 4

calculateCnSE, 6
calculateDistance, 7
calculateDistance, HmmOptions-method
(*HmmOptions-class*), 1
calculateDistance, SnpLevelSet-method
(*calculateDistance*), 7

calls.emission, 9
calls.ICE, 1, 8, 9
calls.ICE, HmmOptions-method
(*HmmOptions-class*), 1
calls.ICE<- (*calls.ICE*), 8
calls.ICE<- , HmmOptions, logical-method
(*HmmOptions-class*), 1

chromosome, chromosome-method
(*HmmPredict-class*), 4
chromosome, HmmPredict-method
(*HmmPredict-class*), 4

chromosome1, 9, 13
chromosomeAnnotation, 15
class.AnnotatedDataFrame, 5

copyNumber.emission, 11
copyNumber.ICE, 1, 10, 11
copyNumber.ICE, HmmOptions-method
(*HmmOptions-class*), 1

copyNumber.ICE<-
(*copyNumber.ICE*), 10

copyNumber.ICE<- , HmmOptions, logical-method
(*HmmOptions-class*), 1

- copyNumber.location, 1
- copyNumber.location
 - (*HmmOptions-methods*), 2
- copyNumber.location, *HmmOptions*, numeric-method
 - (*HmmOptions-class*), 1
- copyNumber.location, *HmmOptions*-method
 - (*HmmOptions-class*), 1
- copyNumber.location<-
 - (*HmmOptions-methods*), 2
- copynumberEmission, 12

- db, *HmmOptions*-method
 - (*HmmOptions-class*), 1

- emission, 12
- emission, *HmmParameter*-method
 - (*HmmParameter-class*), 3
- emission<- (*emission*), 12
- emission<-, *HmmParameter*, array-method
 - (*HmmParameter-class*), 3

- fData, *HmmPredict*-method
 - (*HmmPredict-class*), 4
- featureData, *HmmPredict*-method
 - (*HmmPredict-class*), 4
- featureNames, *HmmPredict*-method
 - (*HmmPredict-class*), 4
- findBreaks, 6, 13
- fit, 13

- genomicDistance, 14, 18
- genomicDistance, *HmmParameter*-method
 - (*HmmParameter-class*), 3
- genomicDistance<-
 - (*genomicDistance*), 14
- genomicDistance<-, *HmmParameter*, numeric-method
 - (*HmmParameter-class*), 3
- genotypeEmission, 14
- getChromosomeArm, 15
- getPar, *HmmPredict*-method
 - (*HmmPredict-class*), 4

- hmm, 5, 16
- hmm, *HmmOptions*, *HmmParameter*-method
 - (*HmmOptions-class*), 1
- HmmOptions*-class, 3, 4, 13
- HmmOptions*-class, 1
- HmmOptions*-methods, 2
- HmmParameter*-class, 2, 13, 16–18
- HmmParameter*-class, 3
- HmmPredict*-class, 16
- HmmPredict*-class, 4
- initialize, *HmmOptions*-method
 - (*HmmOptions-class*), 1
- initialize, *HmmParameter*-method
 - (*HmmParameter-class*), 3
- initialize, *HmmPredict*-method
 - (*HmmPredict-class*), 4
- notes, *HmmOptions*-method
 - (*HmmOptions-class*), 1

- physicalDistance
 - (*calculateDistance*), 7
- physicalDistance, *HmmOptions*-method
 - (*HmmOptions-class*), 1
- physicalDistance, *SnplLevelSet*-method
 - (*calculateDistance*), 7
- Pi, 17
- Pi, *HmmParameter*-method
 - (*HmmParameter-class*), 3
- position, *HmmPredict*-method
 - (*HmmPredict-class*), 4
- predictions (*HmmPredict-class*), 4
- predictions, *HmmPredict*-method
 - (*HmmPredict-class*), 4
- predictions<- (*HmmPredict-class*), 4
- predictions<-, *HmmPredict*, matrix-method
 - (*HmmPredict-class*), 4
- probHomCall (*HmmOptions-methods*), 2
- probHomCall, *HmmOptions*-method
 - (*HmmOptions-class*), 1
- probHomCall<-
 - (*HmmOptions-methods*), 2
- probHomCall<-, *HmmOptions*, numeric-method
 - (*HmmOptions-class*), 1
- sampleNames, *HmmPredict*-method
 - (*HmmPredict-class*), 4
- scaleTransitionProbability, 17, 19
- show, *HmmOptions*-method
 - (*HmmOptions-class*), 1
- show, *HmmParameter*-method
 - (*HmmParameter-class*), 3
- show, *HmmPredict*-method
 - (*HmmPredict-class*), 4
- SnplClass*, *HmmPredict*-method
 - (*HmmPredict-class*), 4
- SnplLevelSet*-class, 16
- snpset (*HmmOptions-methods*), 2
- snpset, *HmmOptions*-method
 - (*HmmOptions-class*), 1
- snpset<- (*HmmOptions-methods*), 2

snpset<-, HmmOptions, SnpLevelSet-method
(*HmmOptions-class*), 1

states, 18

states, HmmOptions-method
(*HmmOptions-class*), 1

states, HmmParameter-method
(*HmmParameter-class*), 3

states, HmmPredict-method
(*HmmPredict-class*), 4

states<- (*states*), 18

states<-, HmmOptions, character-method
(*HmmOptions-class*), 1

states<-, HmmParameter, character-method
(*HmmParameter-class*), 3

summary, HmmPredict-method
(*HmmPredict-class*), 4

tau (*genomicDistance*), 14

tau, HmmParameter-method
(*HmmParameter-class*), 3

term5 (*HmmOptions-methods*), 2

term5, HmmOptions-method
(*HmmOptions-class*), 1

term5<- (*HmmOptions-methods*), 2

term5<-, HmmOptions, numeric-method
(*HmmOptions-class*), 1

transitionScale, 14, 18, 19

transitionScale, HmmParameter-method
(*HmmParameter-class*), 3

transitionScale<-
(*transitionScale*), 19

transitionScale<-, HmmParameter, matrix-method
(*HmmParameter-class*), 3

viterbi, 8, 19