VanillaICE

October 5, 2010

`hmm` *Application of the Viterbi algorithm to copy number and/or genotype data.*

Description

A wrapper for fitting the HMM.

Usage

```
hmm(object, hmm.params, ...)
viterbi(object, hmm.params, verbose=TRUE, normal2altered=1, altered2normal=1,
altered2altered=1, TAUP=1e8)
```

Arguments

`object`	one of the following classes derived from eSet: `SnpSet`, `oligoSnpSet`, `CopyNumberSet`, `CNSet`
`hmm.params`	List. See `hmm.setup`
`verbose`	Logical. Whether to display all messages and warnings.
`normal2altered`	Numeric. Factor for scaling the probability of transitioning from the normal state to an altered state
`altered2normal`	Numeric. Factor for scaling the probability of transitioning from an altered state to a normal state.
`altered2altered`	Numeric. Factor for scaling the probability of transitioning from an altered state to a different altered state
`TAUP`	Numeric. Factor for scaling the distance weighted transition probability. The transition probability is computed as `exp(-2 * d/TAUP)`, where d is the distance between two loci. The default value is 1e8, but users can adjust this number to control the smoothness of the resulting HMM.
`...`	Additional arguments to `viterbi`.

Value

An object of class `RangedData`.

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

`hmm.setup`

Examples

```
data(locusLevelData)
oligoSet <- new("oligoSnpSet",
copyNumber=log2(locusLevelData[["copynumber"]]/100),
call=locusLevelData[["genotypes"]],
callProbability=locusLevelData[["crlmmConfidence"]],
annotation=locusLevelData[["platform"]])
oligoSet <- oligoSet[order(chromosome(oligoSet), position(oligoSet)),]
hmmOpts <- hmm.setup(oligoSet,
     copynumberStates=log2(c(1, 2, 2, 3)),
     states=c("hem-del", "ROH", "normal", "amp"),
     normalIndex=3,
     log.initialP=rep(log(1/4), 4),
     prGenotypeHomozygous=c(0.99, 0.99, 0.7, 0.7))
fit <- hmm(oligoSet, hmmOpts, TAUP=5e7)

## Useful accessors for RangedData
tmp <- ranges(fit)
start(fit)
end(fit)

##Log likelihood ratio comparing likelihood of predicted state to the 'normal' state
## for each segment
fit$LLR
## the number of SNPs / nonpolymorphic loci in each segment
fit$numMarkers

## Expand to a locus x sample matrix
```

`hmm.setup` *Wrapper for computing emission and transition probabilities needed for fitting the hidden Markov model.*

Description

Computes emission probabilities and transition probabilities. See details.

Usage

```
hmm.setup(object, states = paste("state", 1:length(copynumberStates),
sep = ""), ICE = FALSE, copyNumber = TRUE, copynumberStates =
0:4, EMIT.THR = -10, scaleSds = TRUE, verbose = TRUE,
log.initial = log(rep(1/length(states), length(states))),
normalIndex = 3, prGenotypeHomozygous = numeric(),
prGenotypeMissing = rep(1/length(states), length(states)),
pHetCalledHom = 0.001, pHetCalledHet = 0.995, pHomInNormal =
0.8, pHomInRoh = 0.999, rohStates = logical(), trioHmm = FALSE, ...)
```

Arguments

`object`	The object must be one of the following classes inherited from `eSet` and defined in the R package oligoClasses: `SnpSet`, `oligoSnpSet`, `CopyNumberSet`, or `CNSet`.
`states`	Vector of names for the hidden states.
`ICE`	If the object is of class `oligoSnpSet` or `SnpSet` and the R package crlmm was used to call genotypes, the computed emission probabilities incorporate the confidence estimates of the genotype calls.
`copyNumber`	Logical. Whether to include information on copy number in the hidden Markov model. If the object if of class `SnpSet`, this argument is set to FALSE.
`copynumberStates`	Numerical vector with same length as the number of `states`. Each value corresponds to the latent copy number of the hidden state. Note that copynumberStates must be specified on the appropriate scale. If the copy number estimates have been log-transformed, the copynumberStates must be provided on the log-scale.
`EMIT.THR`	Single point outliers can cause the HMM to be jumpy. Emission probabilities below EMIT.THR are set to EMIT.THR.
`scaleSds`	Logical. For objects of class `CNSet`, sd estimates for total copy number are obtained by using `robustSds` function.
`verbose`	Logical. Verbose output during calculations.
`log.initial`	Numeric vector of initial state probabilities (log-scale) corresponding to the hidden states. Must be the same length as `states`.
`normalIndex`	Integer. `states[normalIndex]` should return the name of the 'normal' hidden state, which in general corresponds to copy number 2. For instance, if the states were "hemizygousDeletion", "normal", and "amplification", normalIndex is 2.

`prGenotypeHomozygous`
: Numeric. The probability of a homozygous genotype call in each of the hidden states. Ignored if ICE is TRUE.

`prGenotypeMissing`
: Numeric. The probability of a missing genotype for each hidden states.

`pHetCalledHom`
: Numeric. Probability of misclassifying a genotype call as heterozygous if the true genotype is homozygous. Ignored unless ICE is TRUE.

`pHetCalledHet`
: Numeric. Probability of correctly classifying a genotype call as heterozygous. Ignored unless ICE is TRUE.

`pHomInNormal`
: Numeric. Probability of a homozygous genotype in a region without loss of heterozygosity. Ignored unless ICE is TRUE.

`pHomInRoh`
: Numeric. Probability of a homozygous genotype in a 'region of homozygosity'. Ignored unless ICE is TRUE.

`rohStates`
: Logical vector. TRUE corresponds to a hidden states in which regions of homozygosity are expected. For instance, regions of homozygosity would be TRUE for hidden states corresonding to copy-neutral region of homozygosity (as my occur in a loss of heterozygosity region) and hemizygous deletions.

`trioHmm`
: Logical. This option is experimental. For Father-Mother-Offspring trios, we compute emission probabilities for biparental inheritance where the genotypes are informative. The hidden states correspond to biparental inheritance or non-biparental inheritance. Regions of non-biparental inheritance can be used to quickly flag regions that are possibly de-novo deletions.

`...`
: Ignored.

Details

Details on the calculation of emission probabilities.

Author(s)

R. Scharpf

See Also

`robustSds`

`locusLevelData` *Basic data elements required for the HMM*

Description

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

Usage

```
data(locusLevelData)
```

Format

A list

Details

The basic assay data elements that can be used for fitting the HMM are:

1. a mapping of platform identifiers to chromosome and physical position

2. (optional) a matrix of copy number estimates

3. (optional) a matrix of confidence scores for the copy number estimates (e.g., inverse standard deviations)

4. (optional) a matrix of genotype calls

5. (optional) CRLMM confidence scores for the genotype calls

At least (2) or (4) is required. The locusLevelData is a list that contains (1), (2), (4), and (5).

Source

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

Examples

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

---

robustSds *Calculate robust estimates of the standard deviation*

Description

Uses the median absolute deviation (MAD) to calculate robust estimates of the standard deviation

Usage

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

Arguments

x	A matrix of copy number estimates. Rows are features, columns are samples.
takeLog	Whether to log-transform the copy number estimates before computing robust sds
...	additional arguments to rowMedians

Details

For matrices x with 4 or more samples, the row-wise MAD (SNP-specific sds) are scaled by sample MAD / median(sample MAD).

If the matrix has 3 or fewer samples, the MAD of the sample(s) is returned.

Value

Matrix of standard deviations.

Examples

```
data(locusLevelData)
sds <- robustSds(locusLevelData[["copynumber"]]/100,
 takeLog=TRUE)
```

Index