VanillaICE

March 24, 2012

`HmmOptionList-class`
A container for storing options for the hidden Markov model

Description

A container for storing options for the hidden Markov model.

Objects from the Class

Objects can be created by calls of the form `HmmOptionList(object)`.

Slots

`.Data`: Object of class `"list"` ~~

Extends

Class `"list"`, from data part. Class `"vector"`, by class "list", distance 2. Class `"AssayData"`, by class "list", distance 2. Class `"list_or_ffdf"`, by class "list", distance 2. Class `vectorORfactor`, by class "list", distance 3.

Methods

hmm `signature(object = "CNSet", hmm.params = "HmmOptionList")`: ...

hmm `signature(object = "CopyNumberSet", hmm.params = "HmmOptionList")`: ...

hmm `signature(object = "oligoSnpSet", hmm.params = "HmmOptionList")`: ...

hmm `signature(object = "SnpSet", hmm.params = "HmmOptionList")`: ...

Author(s)

R. Scharpf

Examples

```
showClass("HmmOptionList")
```

HmmOptionList *Constructors for HmmOptionList class*

Description

Constructors for HmmOptionList class. An object of the class contains most of the options for fitting the HMM.

Usage

```
HmmOptionList(object, copynumberStates,
      states, ICE = FALSE, is.log = FALSE,
      scaleSds = TRUE,
      log.initialPr = log(rep(1/length(states), length(states))),
      normalIndex, prGtHom,
      prGtMis = rep(1/length(states), length(states)),
      prHetCalledHom = 0.001, prHetCalledHet = 0.995,
      prHomInNormal = 0.8, prHomInRoh = 0.999,
      rohStates, tau = 1e+08, a2n = 1, n2a = 1, a2a = 1,
      verbose = 2L, ...)
hmm.setup(object, ...)
```

Arguments

object	Object of class SnpSet or one of its derivatives.
copynumberStates	numeric. The mean values for each of the hidden states. Ignored if object is not a CopyNumberSet, oligoSnpSet, or CNSet.
states	Character vector indicating the names of the hidden states
ICE	Logical. Whether to use crlmm confidence scores estimates for computing the emission probabilities of the genotype calls. Only supported for some platforms. See details.
is.log	Logical. Whether the copy number estimates in object are on the log scale. See details
scaleSds	Not implemented. [[Factor to scale the standard deviations used to compute the emission probabilities.]]
log.initialPr	Numeric vector of the same length as the number of states. Specifies the initial state probabilities on the log scale.
normalIndex	Integer. Indicates which element of the states vector corresponds to the 'normal' state.
prGtHom	numeric vector in [0,1], Elements correspond to the probability of a homozygous genotype for each state. Ignored for CopyNumberSet objects.
prGtMis	numeric vector in [0,1], Elements correspond to the probability of a missing genotype for each state. Ignored for CopyNumberSet objects.
prHetCalledHom	numeric in [0,1]. The probability of a heterozygous genotype incorrectly called homozygous. Ignored if object is a CopyNumberSet objects or if ICE is FALSE.

prHetCalledHet
: numeric in [0,1], The probability of a heterozygous genotype correctly called heterozygous. Ignored if object is a CopyNumberSet or if ICE is FALSE.

prHomInNormal
: numeric in [0,1]. The probability of a homozygous genotype in a region with copy number two. Ignored if ICE is FALSE or if object is a CopyNumberSet.

prHomInRoh
: numeric in [0,1]. The probability of observing a homozygous genotype when only homozygous genotypes are expected (e.g., hemizygous deletions, copy-neutral regions of homozygosity). Ignored if ICE is FALSE or if object is a CopyNumberSet.

rohStates
: integer vector. Indicates which states only homozygous genotypes are expected. Ignored if ICE is FALSE or if object is a CopyNumberSet.

tau
: Numeric. Factor for scaling the distance weighted transition probability. See details.

a2n
: Integer. Factor for scaling the probability of transitioning from an altered state to a normal state. Default is 1 (recommended).

n2a
: Integer. Factor for scaling the probability of transitioning from the normal state to an altered state. Default is 1 (recommended).

a2a
: Integer. Factor for scaling the probability of transitioning from an altered state to a different altered stateDefault is 1 (recommended).

verbose
: Logical. Whether to display all messages and warnings.

...
: For hmm.setup, additional arguments passed to HmmOptionList constructor. Ignored for HmmOptionList.

Details

Setting ICE to TRUE requires availability of the crlmm confidence scores for HapMap samples. Currently only a few platforms are supported. See VanillaICE:::icePlatforms().

The transition probability is computed as exp(-2 * d/tau), where d is the distance between two loci. The default value for tau 10^(8), but users can adjust this number to control the smoothness of the resulting HMM.

For object classes that do not support copy number estimation (e.g., SnpSet), is.log is ignored. For all other classes, is.log is used to determine the plausible range of copy number estimates. When is.log is TRUE, we assume the range of the copy number estimates is [-10, 2.5]. Otherwise, the range of the copy number estimates is assumed to be [0, 10]. Values more extreme are truncated. The emission probabilities from the HMM are estimated by fitting a Gaussian-Uniform mixture. The uniform component of the mixture model has a support indicated by the above ranges. The Gaussian component has a mean for each state that is provided by the copynumberStates argument. For oligoSnpSet objects, the standard deviation of the Gaussian can be marker- and sample-specific if confidence scores are provided in the cnConfidence assay data element of the oligoSnpSet object. Specifically, 1/cnConfidence is used as the standard deviation. If the cnConfidence slot is not specified, the median absolute deviation of the copy number estimates across autosomal chromosomes is used for the standard deviation of both autosomal and sex chromosome markers.

Value

Object of class HmmOptionList.

Author(s)

Rob Scharpf

See Also

HmmOptionList, oligoSnpSet, SnpSet, SnpSet, CopyNumberSet

Examples

```
if(require("crlmm")){
  data(cnSetExample, package="crlmm")
  hmm.params <- HmmOptionList(cnSetExample)
  hmm.params
}
data(oligoSetExample, package="oligoClasses")
hmm.params <- HmmOptionList(oligoSet)
hmm.params
```

---

centerAutosomesAt *Center estimates of copy number for autosomes.*

Description

Center estimates of copy number for autosomes.

Usage

```
centerAutosomesAt(x, at, ...)
```

Arguments

x	A oligoSnpSet or CopyNumberSet object.
at	numeric. Value at which to center the copy number estimates (e.g., 2).
...	Ignored

Details

The function sweeps out the column median of the autosomal copy number estimates, and adds back a constant given by at.

Value

A object of the same class as x with centered copy number estimates for the autosomal chromosomes. Chromosomes X and Y are not centered.

Author(s)

R. Scharpf

See Also

link{sweep}

cnEmission-methods *Methods for Function 'cnEmission' in Package 'VanillaICE'*

Description

Methods for calculating the emission probability for estimates of total copy number.

Methods

```
signature(object = "CopyNumberSet", stdev = "ANY")
signature(object = "matrix", stdev = "matrix")
signature(object = "oligoSnpSet", stdev = "ANY")
```

See Also

cnEmission

cnEmission *Calculate emission probabilities for total copy number*

Description

Calculate emission probabilities for total copy number from a Uniform-Gaussian mixture.

Usage

```
cnEmission(object, stdev, k = 5, cnStates, is.log, is.snp, normalIndex, verbose
```

Arguments

object	A CopyNumberSet, oligoSnpSet, or matrix.
stdev	A matrix. Ignored unless class of object is matrix. See details
k	Integer. Size of window for running median. A running median of the total copy number is used to estimate the probability that a copy number estimate is an outlier.
cnStates	Numeric or integer. The theoretical or expected copy number for each hidden state.
is.log	Logical. TRUE if the copy number estimates in object are on the log scale.
is.snp	Logical vector indicating which markers are polymorphic (TRUE) and nonpolymorphic (FALSE)
normalIndex	Integer. The index of the 'normal' copy number state
verbose	Logical.
...	Ignored

Details

We calculate the emission probabilities of the total copy number (CN) estimates from a Normal-Uniform mixture. In particular, we assume the CN (suitably transformed) is emitted from a Normal distribution with mean given by `cnStates`. As outliers are common in high-throughput arrays, we allow for unusual values by adding a Uniform component to the mixture model that covers the support of the CN. (The support is determined by whether the CN is on the log scale as indicated by the `is.log` argument). To estimate the probability that CN is an outlier, we calculate CN - CNsmooth where CNsmooth is calculated from a running median with window given by argument `k`. We assume that the difference (CN-CNsmooth) is a mixture of two Normal distributions – copy number estimates that are not outliers should have a Normal distribution with mean zero and standard deviation 'sigma1', whereas outliers follow a Normal distribution with mean zero and standard deviation 'sigma2', sigma2 » sigma1. We estimate the responsibilities for the mixture via EM, and use these values as a marker-specific estimate of the outlier probability. The emission probability is given by

pihat * N(mean copy number state, sd) + (1-pihat) * Unif(MIN, MAX).

Value

Returns an array of the emission probabilities. The dimensions of the array are [feature index, sample index, state index].

Author(s)

R. Scharpf

See Also

`cnEmission-methods`

See `hmm` method estimates emission probabilities and fits the Viterbi algorithm.

See `gtEmission` for estimating the emission probabilities of diallic genotypes for each of the copy number states.

Examples

```
data(oligoSetExample, package="oligoClasses")
oligoSet <- order(oligoSet)
cn.emit <- cnEmission(oligoSet, k=5,
      cnStates=log2(c(0.1, 1, 2, 3, 4)),
      is.log=TRUE,
                      is.snp=isSnp(oligoSet),
      normalIndex=3)
```

---

gtEmission-methods *Methods for calculating the emission probabilities of diallelic genotypes in Package 'VanillaICE'*

Description

Methods for calculating the emission probabilities of diallelic genotypes in Package **VanillaICE**

Methods

```
signature(object = "matrix")
signature(object = "SnpSet")
```

See Also

gtEmission

---

gtEmission *Calculate emission probabilities for genotypes*

Description

Calculate emission probabilities for genotypes diallelic genotypes.

Usage

```
gtEmission(object, hmm.params, gt.conf, is.snp, cdfName, ...)
```

Arguments

object	A object of class SnpSet, oligoSnpSet, or matrix.
hmm.params	A HmmOptionList.
gt.conf	Ignored unless object is of class matrix.
is.snp	Ignored unless object is of class matrix.
cdfName	Ignored unless the argument ICE for the HmmOptionList object is TRUE.
...	Ignored

Details

Currently, there are two main approaches for calculating the emission probabilities for diallelic genotypes. Which approach is implemented depends on the value of ICE in the HmmOptionList object.

When ICE is FALSE (the default), the emission probabilities for the diallelic genotypes are estimated from a Bernoulli(p_s) with p_s denoting the probability of a homozygous genotype ('AA' or 'BB') for each state s. These probabilities can be specified in the constructor for the HmmOptionList class by the argument prGtHom. The corresponding probability for a heterozygous genotype ('AB') is 1-p_s for state s.

For many calling algorithms, the genotypes are not called. If many no-calls occur in a sample, this can indicate problems with the DNA quality. No calls can also arise when there is poor separation of the genotype clusters or when the A and B allele intensities for a sample do not cluster into any of the diallelic genotype clusters. No calls should be indicated by the value NA. We estimate the emission probability of a no-call using a Bernoulli(p2_s) with p2_s denoting the probability of a no call for state s. The numeric vector for p2 can be specified in the constructor for the HmmOptionList class through the argument prGtMis.

When ICE is TRUE, we assume that the genotype calls were obtained from the **crlmm**. This option is only available for a few of the platforms that **crlmm** supports. The **crlmm** provides an estimate of the confidence for each diallelic genotype call. For the HapMap dataset, we assessed

the distribution of the confidence scores when the call was correct versus the distribution when the call was incorrect (using HapMap genotypes as the gold standard). Using these distributions, we estimate the probability that the true diallelic genotype lies in a region of homozygosity (possibly suggesting a hemizygous deletion or a region of homozygosity induced by uniparental disomy) and the probability that the region has a 'normal' proportion of heterozygous genotypes. The constructor `HmmOptionList` has an argument `rohStates` that indicates which of the hidden states we expect homozygosity. See the reference below for additional details regarding the esimation of emission probabilities using the ICE option.

Value

An array. The dimensions are features x samples x states.

Note

Only chromosomes 1-24 supported (23=X and 24 = Y).

Author(s)

R.Scharpf

References

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

See Also

`cnEmission`, `gtEmission-methods`

Examples

```
data(oligoSetExample, package="oligoClasses")
oligoSet <- order(oligoSet)
oligoSet <- oligoSet[chromosome(oligoSet) == 1, ]
##
## Here, the probability of a missing genotype is 5 times as likely
## for a homozygous-deletion than any of the other states
##
hmmOpts <- HmmOptionList(oligoSet, is.log=TRUE)

## Not run:
## ICE is TRUE
hmmOpts <- HmmOptionList(oligoSet, is.log=TRUE, ICE=TRUE)
tryCatch(gtEmission(oligoSet, hmmOpts), error=function(e) "Annotation platform not suppor
annotation(oligoSet)

## for illustration
annotation(oligoSet) <- "genomewidesnp6"
gt.emit2 <- gtEmission(oligoSet, hmmOpts)
fit1 <- hmm(oligoSet, hmmOpts)
xyplot(cn ~ x | range, data=oligoSet, range=fit1, frame=2e6,
       panel=xypanel, cex=0.5, pch=21, border="orange",
       scales=list(x="free"))

## End(Not run)
```

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, use.baf=FALSE, ...)
```

Arguments

object	one of the following classes derived from eSet: SnpSet, oligoSnpSet, CopyNumberSet, CNSet
hmm.params	HmmOptionList. See details.
use.baf	Logical. Whether to use the BAFs instead of the genotype calls and confidence scores to estimate the emission probabilities. See details.
...	The argument k to the function runmed can be passed for assessing the probability of an outlier. See details.

Details

The probability that a point estimate is an outlier is assessed by subtracting the running median of the copy number estimates from the point estimates (referred to as 'delta'). For copy number changes that span multiple markers, we expect delta to be near zero and to have a small variance. For single point outliers, delta can be large in absolute value and have a large variance. We estimate the mixture probabilities (the probability that an observation is an outlier) using EM. We plug in the estimate of the outlier probability into the emission probabilities. Specifally, we model the emission probabilities as a Uniform-Gaussian mixture, where the Uniform component handles outliers.

The function used for the running median is runmed. One can pass an integer value of k to runmed. Larger values of k should result in fewer segments with low coverage.

For oligoSnpSet objects, the emission probability for the HMM is calculated as

log emission = log GT + log CN

where log GT is the emission probability for the genotype calls and log CN is the emission probability for the copy number estimates. When use.baf is TRUE, the emission probabilities are estimated from the B allele frequencies instead of the genotype calls / confidence scores. Generally, use of the use.baf argument requires a CNSet object obtained from preprocessing with the crlmm package. However, users may supply the B allele frequencies from external sources (e.g., BeadStudio software) when available.

See the examples below for using the use.baf argument and how to visualize the predicted states along with the low-level genotype and copy number summaries.

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, runmed, calculateRBaf

Examples

```
data(locusLevelData, package="oligoClasses")
oligoSet <- new("oligoSnpSet",
copyNumber=log2(locusLevelData[["copynumber"]]/100),
call=locusLevelData[["genotypes"]],
callProbability=locusLevelData[["crlmmConfidence"]],
annotation=locusLevelData[["platform"]])
oligoSet <- oligoSet[!is.na(chromosome(oligoSet)), ]
oligoSet <- oligoSet[order(chromosome(oligoSet), position(oligoSet)),]
oligoSet <- oligoSet[chromosome(oligoSet) < 3, ]
hmmOpts <- hmm.setup(oligoSet, is.log=TRUE)
fit <- hmm(oligoSet, hmmOpts, k=3)
xyplot(cn~x, oligoSet, range=fit[4, ], frame=2e6,pch=21, cex=0.5, panel=xypanel)

## Useful accessors for RangedData
ranges(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
coverage2(fit)

sampleNames(fit)
chromosome(fit)

##---------------------------------------------------------------------------
##
##  Downstream of CRLMM for genotyping
##
##---------------------------------------------------------------------------
## Not run:
if(require("crlmm")){
data(cnSetExample, package="crlmm")
        cnSetExample <- order(cnSetExample)
oligoSet <- as(cnSetExample, "oligoSnpSet")
        assayDataElement(oligoSet, "baf") <- calculateRBaf(cnSetExample)[["baf"]]
        ## uses genotype emission probabilities
        hmmOpts <- HmmOptionList(oligoSet, is.log=TRUE, ICE=FALSE)
        fit2 <- hmm(oligoSet, hmmOpts, use.baf=FALSE)
        xyplot(cn ~ x | range, data=oligoSet, range=fit2[1:10, ], frame=2e6,
               panel=xypanel, cex=0.3, pch=21, border="blue",
               scales=list(x="free"),
```

```
               col.hom="lightblue", col.het="salmon", col.np="grey60", fill.np="grey60")
        xyplot(baf ~ x | range, data=oligoSet, range=fit2[1:10, ], frame=2e6,
               panel=xypanel, cex=0.3, pch=21, border="blue",
               scales=list(x="free"),
               col.hom="lightblue", col.het="salmon", col.np="grey60", fill.np="grey60")
hmmOpts <- HmmOptionList(oligoSet, is.log=TRUE)

        ## use BAFs for emission probs.
        fit3 <- hmm(oligoSet, hmmOpts, use.baf=TRUE)
        ## plot the copy number for the first 10 ranges
        xyplot(cn ~ x | range, data=oligoSet, range=fit3[1:10, ], frame=2e6,
               panel=xypanel, cex=0.2, pch=21, border="blue",
               scales=list(x="free"),
               col.hom="lightblue", col.het="salmon", col.np="grey60", fill.np="grey60")
        ## plot the BAFs for the first 10 ranges
        xyplot(baf ~ x | range, data=oligoSet, range=fit3[1:10, ], frame=2e6,
               panel=xypanel, cex=0.2, pch=21, border="orange",
               scales=list(x="free"),
               col.hom="lightblue", col.het="salmon", col.np="grey60", fill.np="grey60")
} ## if(require("crlmm"))

## End(Not run) ## \dontrun
```

`hmmResults` *Example output from hmm*

Description

Example output from `hmm` method applied to simulated data.

Usage

```
data(hmmResults)
```

Format

A `RangedDataHMM` object.

Details

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

See Also

`xyplot`

Examples

```
data(oligoSetExample, package="oligoClasses")
oligoSet <- oligoSet[chromosome(oligoSet) == 1, ]
hmmOpts <- hmm.setup(oligoSet, is.log=TRUE)
## Not run:
hmmResults <- hmm(oligoSet, hmmOpts, k=3)
```

```
save(hmmResults, file="~/Software/VanillaICE/data/hmmResults.rda")

## End(Not run)
data(hmmResults)
state(hmmResults)
hmmOpts$states[state(hmmResults)]
##
## Plot region of homozygosity
##  (Note that the formula cn~x is required at this time)
xyplot(cn~x, oligoSet, range=hmmResults[2, ], frame=10e6,
       panel=xypanel, pch=21, cex=0.6)
##
## Or, plot each range in its own panel with a frame of 2e6 bases.
##  (Again, the formula is a standard format with cn, x,
##  range, and id the only allowed terms)
## Because these are all the ranges from one individual's
## chromosome, the ranges are overlapping
##  The range 'in focus' is demarcated by vertical blue lines
xyplot(cn~x | range, oligoSet, range=hmmResults, frame=2e6,
       panel=xypanel, pch=21, cex=0.2,
       scales=list(x="free"),
       border="blue",
       col.np="grey",
       col.het="grey",
       col.hom="grey")
```

`icePlatforms` *List platforms for which ICE option is supported.*

Description

Lists platforms for which ICE option is supported.

Usage

```
icePlatforms()
```

Details

When procecessing 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.

Value

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

Author(s)

R. Scharpf

References

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

See Also

HmmOptionList, gtEmission

Examples

```
icePlatforms()
```

---

oligoSnpSet-methods

Methods for oligoSnpSet class

Description

Methods for the oligoSnpSet class

Methods

Standard generic methods:

order(...) If first element in ... is a oligoSnpSet object, the object is ordered by chromosome and physical position.

sd(x) Estimate copy number standard deviation. See sd.

Author(s)

R. Scharpf

See Also

sd, cnEmission, gtEmission

Examples

```
showClass("oligoSnpSet")
```

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

sd-methods *Methods for estimating copy number standard deviations.*

Description

Estimate the standard deviation for CopyNumberSet and oligoSnpSet objects.

Usage

```
sd(x, na.rm=FALSE)
```

Arguments

x	A CopyNumberSet or oligoSnpSet
na.rm	Logical.

Details

The sd method for CopyNumberSet and oligoSnpSet objects retrieves the copy number confidence scores from the cnConfidence assay data element. The confidence matrix is a R x C matrix for an object with R features and C samples. Valid confidence estimates must be positive and not missing (not NA). If any elements in the confidence matrix are invalid, a robust estimate of the standard deviation is computed (described below). If all elements are valid, the standard deviation matrix is returned as 1 / confidence.

If any elements in the confidence matrix are invalid, the standard deviation for each marker and sample is calculated as follows. If autosomal markers are present, the standard deviation is estimated as the median absolute deviation across autosomal markers for each sample. This gives a vector of length C. The R x C standard deviation matrix is populated by row from the vector of length C (the standard deviation for each marker in a sample is given the same standard deviation). If autosomal markers are not present, the median absolute deviation across X-chromosome markers and Y-chromosome markers are estimated independently, providing to vectors of length C. The matrix of standard deviations for the X chromosome is populated by the C-length vector for the X-chromosome (by-row) and likewise for the Y chromosome.

Value

A matrix.

See Also

mad

Examples

```
data(oligoSetExample)
sds <- sd(oligoSet)
```

---

xypanel *A panel function for plotting copy number versus physical position*

Description

A panel function for xyplot for plotting copy number versus physical position.

Usage

```
xypanel(x, y, gt, is.snp, range, col.hom = "grey20", fill.hom =
"lightblue", col.het = "grey20", fill.het = "salmon", col.np = "grey20",
fill.np = "grey60", show.state=TRUE, ..., subscripts)
```

Arguments

x	Physical position in megabases.
y	Copy number estimates.
gt	Genotype calls.
is.snp	Logical. Whether the marker is polymorphic.
range	A RangedData or IRanges object. Note that we expect the units returned by start and end to be basepairs.
col.hom	A specification for the color of plotting symbols for homozygous genotypes.
fill.hom	A specification for the fill color of plotting symbols for homozygous genotypes.
col.het	A specification for the color of plotting symbols for heterozygous genotypes.
fill.het	A specification for the fill color of plotting symbols for heterozygous genotypes.
col.np	A specification for the color of plotting symbols for nonpolymorphic markers.
fill.np	A specification for the fill color of plotting symbols for nonpolymorphic genotypes.
show.state	Logical. Whether to display the predicted state in each panel.
...	Additional arguments passed to lattice functions xyplot, lpoints, and lrect.
subscripts	

Details

The order of plotting is (1) nonpolymorphic markers, (2), homozygous SNPs, and (3) heterozygous SNPs. Stretches of homozygosity should appear as blue using the default color scheme.

Note

To make the drawing of the range object border invisible, one can use border="white".

Author(s)

R. Scharpf

See Also

xyplot

Examples

```
## Not run:
if(require("crlmm")){
data(cnSetExample, package="crlmm")
        cnSetExample <- order(cnSetExample)
oligoSet <- as(cnSetExample, "oligoSnpSet")
        ## uses genotype emission probabilities
        hmmOpts <- HmmOptionList(oligoSet, is.log=TRUE, ICE=FALSE)
        fit2 <- hmm(oligoSet, hmmOpts, use.baf=FALSE)
        xyplot(cn ~ x | range, data=oligoSet, range=fit2[1:10, ], frame=2e6,
               panel=xypanel, cex=0.3, pch=21, border="blue",
               scales=list(x="free"),
               col.hom="lightblue", col.het="salmon", col.np="grey60",
  fill.np="grey60")
```

```
library(lattice)
## now the generic for xyplot is masked.  Must do
xyplot <- VanillaICE:::xyplot

}


## End(Not run)
```

`xyplot` *Plot copy number and physical position for a set of genomic intervals.*

Description

Plot copy number and physical position given by a `CNSet` object for a set of genomic intervals stored in a `RangedDataCVN` object.

Usage

```
xyplot(x, data, ...)
xyplot2(x, data, range, frame=50e3L, ...)
```

Arguments

`x`	A `formula`. Currently, the formula must be one of `cn~x`, `cn ~ x | id` or `cn ~ x | range`.
`data`	A `CNSet` or `SnpSet` object.
`...`	A `RangedDataCNV` object must be passed by the name 'range'. Arguments for `xyplot` are passed to `xyplot2`. Additional arguments are passed to `xypanel` and `panel.xyplot`.
`range`	A `RangedDataCNV` object.
`frame`	The genomic distance (basepairs) to the left and right of the start and stop coordinates in the `range` object.

Details

For a given `RangedDataCNV` object, this function will plot the copy number estimates versus physical position. The function is particularly useful for multi-panel displays in which the copy number estimates for a single range of the `RangedDataCNV` object appears in one panel. The size of the multi-panel display depends on the number of ranges (rows) in the `RangedDataCNV` object. Typically, one would want to pass no more than 10 ranges to the `xyplot` function.

For genomic intervals of interest in the `RangedDataCNV`, it is often helpful to 'frame' the interval by plotting the data surrounding the interval. To facilitate this process, one may pass an argument called `frame` (an integer) that indicates the number of basepairs to the left and right of the start / stop points in the interval. By default, the first interval in the `RangedDataCNV` object will be plotted in the lower left panel and the last interval `RangedDataCNV` object will be plotted in the upper right panel. Overplotting the copy number data in each panel is a rectangle that indicates the start and stop coordinates in the `RangedDataCNV` object.

Value

An object of class `trellis`.

Note

Note that users must pass a `RangedDataCNV` object called 'range'. As mentioned previously, it can be helpful to pass an integer called 'frame' that indicates how much contextual data we should plot surrounding each genomic interval.

If the `lattice` package is loaded after loading `VanillaICE`, the generic definition for `xyplot` in `VanillaICE` will be masked. To unmask the S4 generic in `VanillaICE`, do

xyplot <- VanillaICE:::xyplot

Author(s)

R. Scharpf

See Also

`xyplot`, `xypanel`

To modify the plot appearance from the default, additional arguments can be passed to `panel.xyplot`, `lpoints`, and `lrect`.

Examples

```
## simulated data
data(oligoSetExample, package="oligoClasses")
data(hmmResults)
## to visualize each range in it's own panel surrounded by a
## frame of 2,000,000 bases:
## (here the frames are overlapping, but the method could be
## applied more generally to a collection of ranges from
## different chromsomes and samples)
xyplot(cn ~ x | range, data=oligoSet, range=hmmResults, frame=2e6,
       panel=xypanel, cex=0.5, pch=21, border="orange",
       scales=list(x="free"))

   if(require("lattice")){
        ## now the generic for xyplot in VanillaICE is masked.  To unmask,
        xyplot <- VanillaICE:::xyplot
   }
```

Index

∗Topic **array**
cnEmission, 5
gtEmission, 7
∗Topic **classes**
HmmOptionList-class, 1
oligoSnpSet-methods, 13
∗Topic **color**
xypanel, 15
∗Topic **datasets**
hmmResults, 11
∗Topic **distribution**
cnEmission, 5
gtEmission, 7
∗Topic **dplot**
xypanel, 15
xyplot, 17
∗Topic **list**
HmmOptionList, 2
∗Topic **manip**
centerAutosomesAt, 4
hmm, 9
robustSds, 14
∗Topic **methods**
cnEmission-methods, 5
gtEmission-methods, 6
sd-methods, 14
xyplot, 17
∗Topic **misc**
icePlatforms, 12
∗Topic **models**
hmm, 9
∗Topic **ts**
hmm, 9

AssayData, *1*

calculateRBaf, *10*
centerAutosomesAt, 4
cnEmission, *5*, 5, *8*, *13*
cnEmission,CopyNumberSet,ANY-method
(cnEmission-methods), 5
cnEmission,matrix,matrix-method
(cnEmission-methods), 5
cnEmission,oligoSnpSet,ANY-method
(cnEmission-methods), 5
cnEmission-methods, *6*
cnEmission-methods, 5
CopyNumberSet, *4*

gtEmission, *6*, *7*, 7, *13*
gtEmission,matrix-method
(gtEmission-methods), 6
gtEmission,SnpSet-method
(gtEmission-methods), 6
gtEmission-methods, *8*
gtEmission-methods, 6

hmm, *6*, 9
hmm,CNSet,HmmOptionList-method
(HmmOptionList-class), 1
hmm,CopyNumberSet,HmmOptionList-method
(HmmOptionList-class), 1
hmm,oligoSnpSet,HmmOptionList-method
(HmmOptionList-class), 1
hmm,SnpSet,HmmOptionList-method
(HmmOptionList-class), 1
hmm.setup, *10*
hmm.setup *(HmmOptionList)*, 2
HmmOptionList, 2, *4*, *13*
HmmOptionList-class, 1
hmmResults, 11

icePlatforms, 12

list, *1*
list_or_ffdf, *1*
lpoints, *18*
lrect, *18*

mad, *15*

oligoSnpSet, *4*
oligoSnpSet-methods, 13
order,oligoSnpSet-method
(oligoSnpSet-methods), 13

panel.xyplot, *18*

robustSds, 14
runmed, *10*

sd, *13*
sd *(sd-methods)*, 14
sd,CopyNumberSet-method
(sd-methods), 14
sd,oligoSnpSet-method
(oligoSnpSet-methods), 13
sd-methods, 14
SnpSet, *4*

vector, *1*

xypanel, 15, *18*
xyplot, *11*, *16*, 17, *18*
xyplot,formula,SnpSet-method
(xyplot), 17
xyplot2 *(xyplot)*, 17
xyplot2,formula,CNSet,RangedDataCNV-method
(xyplot), 17
xyplot2,formula,eSet,RangedDataCNV-method
(xyplot), 17