crlmm

April 20, 2011

`AssayData-methods` *Methods for class "AssayData" in crlmm*

Description

The `batchStatistics` slot in a `CNSet` object is an instance of the `AssayData` slot. In general, the accessors for `AssayData` are called indirectly by the corresponding method for the `CNSet` class and not called directly by the user.

Methods

Ns `signature(object="AssayData")`: ...

corr `signature(object="AssayData")`: ...

mads `signature(x="AssayData")`: ...

medians `signature(object="AssayData")`: ...

tau2 `signature(object="AssayData")`: ...

See Also

`CNSet-class`, `Ns`, `tau2`, `corr`, `mads`, `medians`

`batchStatisticAccessors`
Accessors for batch-specific summary statistics.

Description

The summary statistics stored here are used by the tools for copy number estimation.

Usage

```
corr(object, ...)
tau2(object, ...)
mads(object,...)
medians(object,...)
Ns(object,...)
```

Arguments

object	An object of class CNSet.
...	An additional argument named 'i' can be passed to subset the markers and an argument 'j' can be passed to subset the batches. Other arguments are ignored.

Value

An array with dimension R x A x G x C, or R x G x C.

R: number of markers A: number of alleles (2) G: number of biallelic genotypes (3) C: number of batches

Ns returns an array of genotype frequencies stratified by batch. Dimension R x G x C.

corr returns an array of within-genotype correlations (log2-scale) stratified by batch. Dimension R x G x C.

medians returns an array of the within-genotype medians (intensity-scale) stratified by batch and allele. Dimension R x A x G x C.

mads returns an array of the within-genotype median absolute deviations (intensity-scale) stratified by batch and allele. Dimension is the same as for medians.

tau2 returns an array of the squared within-genotype median absolute deviation on the log-scale. Only the mads for AA and BB genotypes are stored. Dimension is R x A x G x C, where G is AA or BB. Note that the mad for allele A/B for subjects with genotype BB/AA is a robust estimate of the background variance, whereas the the mad for allele A/B for subjects with genotype AA/BB is a robust estimate of the variance for copy number greater than 0 (we assume that on the log-scale the variance is rougly constant for CA, CB > 0).

See Also

batchStatistics

Examples

```
data(sample.CNSetLM)
## update to class CNSet
cnSet <- as(sample.CNSetLM, "CNSet")
## All NAs. Need to replace sample.CNSetLM with a HapMap example
Ns(cnSet, i=1:5, j=1:2)
corr(cnSet, i=1:5, j=1:2)
medians(cnSet, i=1:5, j=1:2)
mads(cnSet, i=1:5, j=1:2)
tau2(cnSet, i=1:5, j=1:2)
```

---

celDates *Extract dates from the cel file header*

Description

Extract dates from the cel file header.

Usage

```
celDates(celfiles)
```

Arguments

`celfiles`	CEL file names. Must specify the complete path.

Value

date-time class POSIXt

Author(s)

R. Scharpf

See Also

read.celfile.header, *POSIXt*

`CNSet-methods` *crlmm methods for class "CNSet"*

Description

CNSet is a container defined in the oligoClasses package for storing normalized intensities for genotyping platforms, genotype calls, and parameters estimated for copy number. Accessors for data that an object of this class contains are largely defined in the package oligoClasses. CNSet methods that involve more complex calculations that are specific to the crlmm package, such as computing allele-specific copy number, are included in crlmm and described here.

Methods

CA `signature(object="CNSet")`: ...

CB `signature(object="CNSet")`: ...

lines `signature(x="CNSet")`: ...

totalCopynumber `signature(object="CNSet")`: ...

nuA `signature(object="CNSet")`: ...

nuB `signature(object="CNSet")`: ...

phiA `signature(object="CNSet")`: ...

phiB `signature(object="CNSet")`: ...

Ns `signature(object="CNSet")`: ...

corr `signature(object="CNSet")`: ...

mads `signature(x="CNSet")`: ...

medians `signature(object="CNSet")`: ...

tau2 `signature(object="CNSet")`: ...

See Also

`CNSet-class`, `CA`, `CB`, `totalCopynumber`

`constructIlluminaCNSet` *Construct an instance of CNSetLM after preprocessing Illumina files*

Description

Assemble the preprocessed data and genotype calls from `crlmmIllumina` to initialize a `CNSetLM` object.

Usage

```
constructIlluminaCNSet(crlmmResult, path, snpFile, cnFile)
```

Arguments

`crlmmResult`	A `SnpSet` object returned by function `crlmmIllumina` or `crlmmIllumina2`.
`path`	path to files created by `crlmmIllumina`
`snpFile`	The `snpFile` filename specified in `crlmmIllumina`.
`cnFile`	The `cnFile` filename specified in `crlmmIllumina`.

Value

An object of class `CNSetLM`.

Author(s)

R. Scharpf

See Also

`CNSet-class`, `crlmmIllumina`

`copynumberAccessors` *Accessors for allele-specific or total copy number*

Description

These methods can be applied after an object of class `CNSet` has been generated by the `crlmmCopynumber` function.

Usage

```
CA(object, ...)
CB(object, ...)
nuA(object)
nuB(object)
phiA(object)
phiB(object)
totalCopynumber(object,...)
```

Arguments

object	An object of class CNSet.
...	An additional argument named 'i' can be passed to subset the markers and an argument 'j' can be passed to subset the samples. Other arguments are ignored.

Details

At polymorphic markers, nuA and nuB provide the intercept coefficient (the estimated background intensity) for the A and B alleles, respectively. phiA and phiB provide the slope coefficients for the A and B alleles, respectively.

At nonpolymorphic markers, nuB and phiB are 'NA'.

These functions can be used to tranlate the normalized intensities to the copy number scale. Plotting the copy number estimates as a function of physical position can be used to guide downstream algorithms that smooth, as well as to assess possible mosaicism.

Value

nu[A/B] and phi[A/B] return matrices of the intercept and slope coefficients, respectively.

CA and CB return matrices of allele-specific copy number.

totalCopynumber returns a matrix of CA+CB.

See Also

crlmmCopynumber, CNSet-class

Examples

```
## Version 1.6* of crlmm used CNSetLM objects.
data(sample.CNSetLM)

## To update to class CNSet, use
cnSet <- as(sample.CNSetLM, "CNSet")
all(isCurrent(cnSet)) ## is the cnSet object current?

## ------------------------------------------------
## calculating allele-specific copy number
## ------------------------------------------------
## copy number for allele A, first 5 markers, first 2 samples
(ca <- CA(cnSet, i=1:5, j=1:2))
## copy number for allele B, first 5 markers, first 2 samples
(cb <- CB(cnSet, i=1:5, j=1:2))
## total copy number for first 5 markers, first 2 samples
(cn1 <- ca+cb)

## total copy number at first 5 nonpolymorphic loci
index <- which(!isSnp(cnSet))[1:5]
cn2 <- CA(cnSet, i=index, j=1:2)
## note, cb is NA at nonpolymorphic loci
(cb <- CB(cnSet, i=index, j=1:2))
## note, ca+cb will give NAs at nonpolymorphic loci
CA(cnSet, i=index, j=1:2) + cb
## A shortcut for total copy number
cn3 <- totalCopynumber(cnSet, i=1:5, j=1:2)
```

```
all.equal(cn3, cn1)
cn4 <- totalCopynumber(cnSet, i=index, j=1:2)
all.equal(cn4, cn2)

## markers 1-5, all samples
cn5 <- totalCopynumber(cnSet, i=1:5)
## all markers, samples 1-5
cn6 <- totalCopynumber(cnSet, j=1:5)

## NOTE: subsetting the object before extracting copy number
##       can be very inefficient when the data set is very large,
##       particularly if using ff objects.  IN particular, subsetting
##       the CNSet object will subset all of the assay data elements
##       and all of the elements in the LinearModelParameter slot
## Not run:
       ## do not do the following
cn <- CA(cnSet[1:5, ], "A")

## End(Not run)
```

`crlmmCopynumber` *Locus- and allele-specific estimation of copy number*

Description

Locus- and allele-specific estimation of copy number.

Usage

```
crlmmCopynumber(object, MIN.SAMPLES=10, SNRMin = 5, MIN.OBS = 1,
        DF.PRIOR = 50, bias.adj = FALSE,
                prior.prob = rep(1/4, 4), seed = 1, verbose = TRUE,
                GT.CONF.THR = 0.95, MIN.NU = 2^3, MIN.PHI = 2^3,
                THR.NU.PHI = TRUE, type=c("SNP", "NP", "X.SNP", "X.NP"))
```

Arguments

`object`	object of class `CNSet`.
`MIN.SAMPLES`	'Integer'. The minimum number of samples in a batch. Bathes with fewer than MIN.SAMPLES are skipped. Therefore, samples in batches with fewer than MIN.SAMPLES have NA's for the allele-specific copy number and NA's for the linear model parameters.
`SNRMin`	Samples with low signal to noise ratios are excluded.
`MIN.OBS`	For a SNP with with fewer than `MIN.OBS` of a genotype in a given batch, the within-genotype median is imputed. The imputation is based on a regression using SNPs for which all three biallelic genotypes are observed. For example, assume at at a given SNP genotypes AA and AB were observed and BB is an unobserved genotype. For SNPs in which all 3 genotypes were observed, we fit the model E(mean_BB) = beta0 + beta1*mean_AA + beta2*mean_AB, obtaining estimates; of beta0, beta1, and beta2. The imputed mean at the SNP with unobserved BB is then beta0hat + beta1hat * mean_AA of beta2hat * mean_AB.

DF.PRIOR	The 2 x 2 covariance matrix of the background and signal variances is estimated from the data at each locus. This matrix is then smoothed towards a common matrix estimated from all of the loci. DF.PRIOR controls the amount of smoothing towards the common matrix, with higher values corresponding to greater smoothing. Currently, DF.PRIOR is not estimated from the data. Future versions may estimate DF.PRIOR empirically.
bias.adj	bias.adj is currently ignored (as well as the prior.prob argument). We plan to add this feature back to the crlmm package in the near future. This feature, when TRUE, updated initial estimates from the linear model after excluding samples with a low posterior probability of normal copy number. Excluding samples that have a low posterior probability can be helpful at loci in which a substantial fraction of the samples have a copy number alteration. For additional information, see Scharpf et al., 2010.
prior.prob	This argument is currently ignored. A numerical vector providing prior probabilities for copy number states corresponding to homozygous deletion, hemizygous deletion, normal copy number, and amplification, respectively.
seed	Seed for random number generation.
verbose	Logical.
GT.CONF.THR	Confidence threshold for genotype calls (0, 1). Calls with confidence scores below this theshold are not used to estimate the within-genotype medians. See Carvalho et al., 2007 for information regarding confidence scores of biallelic genotypes.
MIN.NU	numeric. Minimum value for background intensity. Ignored if THR.NU.PHI is FALSE.
MIN.PHI	numeric. Minimum value for slope. Ignored if THR.NU.PHI is FALSE.
THR.NU.PHI	If THR.NU.PHI is FALSE, MIN.NU and MIN.PHI are ignored. When TRUE, background (nu) and slope (phi) coefficients below MIN.NU and MIN.PHI are set to MIN.NU and MIN.PHI, respectively.
type	Character string vector that must be one or more of "SNP", "NP", "X.SNP", or "X.NP". Type refers to a set of markers. See details below

Details

We suggest a minimum of 10 samples per batch for using crlmmCopynumber. 50 or more samples per batch is preferred and will improve the estimates.

The function crlmmCopynumber uses matrices instead of ff objects if the ff library is not loaded. When the ff package is loaded, large data support is enabled. Normalized intensities (alleleA and alleleB), genotype calls and confidence scores (snpCall and snpCallProbability) are stored in assayData slot. Summary statistics for each batch, including the linear model paramters for copy number, are stored in the batchStatistics slot. Both the assayData and batchStatistics slot are of class AssayData with elements that are ff objects (if ff package is loaded) or matrices.

The functions crlmmCopynumberLD and crlmmCopynumber2 have been deprecated.

The argument type can be used to specify a subset of markers for which the copy number estimation algorithm is run. One or more of the following possible entries are valid: 'SNP', 'NP', 'X.SNP', and 'X.NP'.

'SNP' referers to autosomal SNPs.

'NP' refers to autosomal nonpolymorphic markers.

'X.SNP' refers to SNPs on chromosome X.

'X.NP' refers to autosomes on chromosome X.

Author(s)

R. Scharpf

References

Carvalho B, Bengtsson H, Speed TP, Irizarry RA. Exploration, normalization, and genotype calls of high-density oligonucleotide SNP array data. Biostatistics. 2007 Apr;8(2):485-99. Epub 2006 Dec 22. PMID: 17189563.

Carvalho BS, Louis TA, Irizarry RA. Quantifying uncertainty in genotype calls. Bioinformatics. 2010 Jan 15;26(2):242-9.

Scharpf RB, Ruczinski I, Carvalho B, Doan B, Chakravarti A, and Irizarry RA, Biostatistics. Biostatistics, Epub July 2010.

`crlmmIllumina` *Genotype Illumina Infinium II BeadChip data with CRLMM*

Description

Implementation of the CRLMM algorithm for data from Illumina's Infinium II BeadChips.

Usage

```
crlmmIllumina(RG, XY, stripNorm=TRUE,
      useTarget=TRUE, row.names=TRUE, col.names=TRUE,
      probs=c(1/3, 1/3, 1/3), DF=6, SNRMin=5,
      gender=NULL, seed=1, mixtureSampleSize=10^5,
      eps=0.1, verbose=TRUE, cdfName, sns, recallMin=10,
      recallRegMin=1000, returnParams=FALSE, badSNP=0.7)
```

Arguments

`RG`	`NChannelSet` containing R and G bead intensities
`XY`	`NChannelSet` containing X and Y bead intensities
`stripNorm`	'logical'. Should the data be strip-level normalized?
`useTarget`	'logical' (only used when `stripNorm=TRUE`). Should the reference HapMap intensities be used in strip-level normalization?
`row.names`	'logical'. Use rownames - SNP names?
`col.names`	'logical'. Use colnames - Sample names?
`probs`	'numeric' vector with priors for AA, AB and BB.
`DF`	'integer' with number of degrees of freedom to use with t-distribution.
`SNRMin`	'numeric' scalar defining the minimum SNR used to filter out samples.
`gender`	'integer' vector, with same length as 'filenames', defining sex. (1 - male; 2 - female)

seed	'integer' scalar for random number generator (used to sample mixtureSampleSize SNPs for mixture model.
mixtureSampleSize	'integer'. The number of SNP's to be used when fitting the mixture model.
eps	Minimum change for mixture model.
verbose	'logical'.
cdfName	'character' defining the chip annotation (manifest) to use ('human370v1c', human550v3b', 'human650v3a', 'human1mv1c', 'human370quadv3c', 'human610quadv1b', 'human660quadv1a', 'human1mduov3b', 'humanomni1quadv1b', 'humanomniexpress12v1b')
sns	'character' vector with sample names to be used.
recallMin	'integer'. Minimum number of samples for recalibration.
recallRegMin	'integer'. Minimum number of SNP's for regression.
returnParams	'logical'. Return recalibrated parameters.
badSNP	'numeric'. Threshold to flag as bad SNP (affects batchQC)

Details

Note: The user should specify either the RG or XY intensities, not both.

Value

A SnpSet object which contains

calls	Genotype calls (1 - AA, 2 - AB, 3 - BB)
callProbability	confidence scores 'round(-1000*log2(1-p))'

in the assayData slot and

SNPQC	SNP Quality Scores
batchQC	Batch Quality Scores

along with center and scale parameters when returnParams=TRUE in the featureData slot.

Author(s)

Matt Ritchie

References

Ritchie ME, Carvalho BS, Hetrick KN, Tavar\'e S, Irizarry RA. R/Bioconductor software for Illumina's Infinium whole-genome genotyping BeadChips. Bioinformatics. 2009 Oct 1;25(19):2621-3.

Carvalho B, Bengtsson H, Speed TP, Irizarry RA. Exploration, normalization, and genotype calls of high-density oligonucleotide SNP array data. Biostatistics. 2007 Apr;8(2):485-99. Epub 2006 Dec 22. PMID: 17189563.

Carvalho BS, Louis TA, Irizarry RA. Quantifying uncertainty in genotype calls. Bioinformatics. 2010 Jan 15;26(2):242-9.

Examples

```
## crlmmOut = crlmmIllumina(RG)
```

`crlmmIlluminaV2` *Read and Genotype Illumina Infinium II BeadChip data with CRLMM*

Description

Implementation of the CRLMM algorithm for data from Illumina's Infinium II BeadChips.

Usage

```
crlmmIlluminaV2(sampleSheet=NULL, arrayNames=NULL, ids=NULL, path=".",
      arrayInfoColNames=list(barcode="SentrixBarcode_A", position="SentrixPositi
      highDensity=FALSE, sep="_", fileExt=list(green="Grn.idat", red="Red.idat")
      saveDate=FALSE, stripNorm=TRUE, useTarget=TRUE, row.names=TRUE, col.names=
      probs=c(1/3, 1/3, 1/3), DF=6, SNRMin=5, gender=NULL,
      seed=1, mixtureSampleSize=10^5, eps=0.1, verbose=TRUE,
      cdfName, sns, recallMin=10, recallRegMin=1000,
      returnParams=FALSE, badSNP=.7)
```

Arguments

`sampleSheet`	`data.frame` containing Illumina sample sheet information (for required columns, refer to BeadStudio Genotyping guide - Appendix A).
`arrayNames`	character vector containing names of arrays to be read in. If `NULL`, all arrays that can be found in the specified working directory will be read in.
`ids`	vector containing ids of probes to be read in. If `NULL` all probes found on the first array are read in.
`path`	character string specifying the location of files to be read by the function
`arrayInfoColNames`	(used when `sampleSheet` is specified) list containing elements 'barcode' which indicates column names in the `sampleSheet` which contains the arrayNumber/barcode number and 'position' which indicates the strip number. In older style sample sheets, this information is combined (usually in a column named 'SentrixPosition') and this should be specified as `list(barcode=NULL, position="SentrixPosition")`
`highDensity`	logical (used when `sampleSheet` is specified). If `TRUE`, array extensions '_A', '_B' in sampleSheet are replaced with 'R01C01', 'R01C02' etc.
`sep`	character string specifying separator used in .idat file names.
`fileExt`	list containing elements 'Green' and 'Red' which specify the .idat file extension for the Cy3 and Cy5 channels.
`saveDate`	'logical'. Should the dates from each .idat be saved with sample information?
`stripNorm`	'logical'. Should the data be strip-level normalized?
`useTarget`	'logical' (only used when `stripNorm=TRUE`). Should the reference HapMap intensities be used in strip-level normalization?
`row.names`	'logical'. Use rownames - SNP names?
`col.names`	'logical'. Use colnames - Sample names?
`probs`	'numeric' vector with priors for AA, AB and BB.

DF	'integer' with number of degrees of freedom to use with t-distribution.
SNRMin	'numeric' scalar defining the minimum SNR used to filter out samples.
gender	'integer' vector, with same length as 'filenames', defining sex. (1 - male; 2 - female)
seed	'integer' scalar for random number generator (used to sample mixtureSampleSize SNPs for mixture model.
mixtureSampleSize	'integer'. The number of SNP's to be used when fitting the mixture model.
eps	Minimum change for mixture model.
verbose	'logical'.
cdfName	'character' defining the chip annotation (manifest) to use ('human370v1c', human550v3b', 'human650v3a', 'human1mv1c', 'human370quadv3c', 'human610quadv1b', 'human660quadv1a', 'human1mduov3b', 'humanomni1quadv1b', 'humanomniexpress12v1b')
sns	'character' vector with sample names to be used.
recallMin	'integer'. Minimum number of samples for recalibration.
recallRegMin	'integer'. Minimum number of SNP's for regression.
returnParams	'logical'. Return recalibrated parameters.
badSNP	'numeric'. Threshold to flag as bad SNP (affects batchQC)

Details

This function combines the reading of data from idat files using readIdatFiles and genotyping to reduce memory usage.

Value

A SnpSet object which contains

calls	Genotype calls (1 - AA, 2 - AB, 3 - BB)
callProbability	confidence scores 'round(-1000*log2(1-p))'

in the assayData slot and

SNPQC	SNP Quality Scores
batchQC	Batch Quality Scores

along with center and scale parameters when returnParams=TRUE in the featureData slot.

Author(s)

Matt Ritchie

References

Ritchie ME, Carvalho BS, Hetrick KN, Tavar\'e S, Irizarry RA. R/Bioconductor software for Illumina's Infinium whole-genome genotyping BeadChips. Bioinformatics. 2009 Oct 1;25(19):2621-3.

Carvalho B, Bengtsson H, Speed TP, Irizarry RA. Exploration, normalization, and genotype calls of high-density oligonucleotide SNP array data. Biostatistics. 2007 Apr;8(2):485-99. Epub 2006 Dec 22. PMID: 17189563.

Carvalho BS, Louis TA, Irizarry RA. Quantifying uncertainty in genotype calls. Bioinformatics. 2010 Jan 15;26(2):242-9.

See Also

crlmmIllumina

Examples

```
## crlmmOut = crlmmIlluminaV2(samples,path=path,arrayInfoColNames=list(barcode="Chip",pos
##                            saveDate=TRUE,cdfName="human370v1c",returnParams=TRUE)
```

`crlmm-package` *Genotype Calling via CRLMM Algorithm*

Description

Faster implementation of CRLMM specific to SNP 5.0 and 6.0 arrays.

Details

Index:

```
crlmm-package           New implementation of the CRLMM Algorithm.
crlmm                   Genotype SNP 5.0 or 6.0 samples.
calls                   Accessor for genotype calls.
confs                   Accessor for confidences.
```

The 'crlmm' package reimplements the CRLMM algorithm present in the 'oligo' package. This implementation primes for efficient genotyping of samples on SNP 5.0 and SNP 6.0 Affymetrix arrays.

To use this package, the user must have additional data packages: 'genomewidesnp5Crlmm' - SNP 5.0 arrays 'genomewidesnp6Crlmm' - SNP 6.0 arrays

Author(s)

Rafael A Irizarry Maintainer: Benilton S Carvalho <carvalho@bclab.org>

References

Carvalho BS, Louis TA, Irizarry RA. Quantifying uncertainty in genotype calls. Bioinformatics. 2010 Jan 15;26(2):242-9. Epub 2009 Nov 11.

`crlmm` *Genotype oligonucleotide arrays with CRLMM*

Description

This is a faster and more efficient implementation of the CRLMM algorithm, especially designed for Affymetrix SNP 5 and 6 arrays (to be soon extended to other platforms).

Usage

```
crlmm(filenames, row.names=TRUE, col.names=TRUE,
      probs=c(1/3, 1/3, 1/3), DF=6, SNRMin=5,
      gender=NULL, save.it=FALSE, load.it=FALSE,
      intensityFile, mixtureSampleSize=10^5,
      eps=0.1, verbose=TRUE, cdfName, sns, recallMin=10,
      recallRegMin=1000, returnParams=FALSE, badSNP=0.7)
crlmm2(filenames, row.names=TRUE, col.names=TRUE,
      probs=c(1/3, 1/3, 1/3), DF=6, SNRMin=5,
      gender=NULL, save.it=FALSE, load.it=FALSE,
      intensityFile, mixtureSampleSize=10^5,
      eps=0.1, verbose=TRUE, cdfName, sns, recallMin=10,
      recallRegMin=1000, returnParams=FALSE, badSNP=0.7)
```

Arguments

`filenames`	'character' vector with CEL files to be genotyped.
`row.names`	'logical'. Use rownames - SNP names?
`col.names`	'logical'. Use colnames - Sample names?
`probs`	'numeric' vector with priors for AA, AB and BB.
`DF`	'integer' with number of degrees of freedom to use with t-distribution.
`SNRMin`	'numeric' scalar defining the minimum SNR used to filter out samples.
`gender`	'integer' vector, with same length as 'filenames', defining sex. (1 - male; 2 - female)
`save.it`	'logical'. Save preprocessed data?
`load.it`	'logical'. Load preprocessed data to speed up analysis?
`intensityFile`	'character' with filename to be saved/loaded - preprocessed data.
`mixtureSampleSize`	Number of SNP's to be used with the mixture model.
`eps`	Minimum change for mixture model.
`verbose`	'logical'.
`cdfName`	'character' defining the CDF name to use ('GenomeWideSnp5', 'GenomeWideSnp6')
`sns`	'character' vector with sample names to be used.
`recallMin`	Minimum number of samples for recalibration.
`recallRegMin`	Minimum number of SNP's for regression.
`returnParams`	'logical'. Return recalibrated parameters.
`badSNP`	'numeric'. Threshold to flag as bad SNP (affects batchQC)

Details

'crlmm2' allows one to genotype very large datasets (via ff package) and also permits the use of clusters or multiple cores (via snow package) to speed up genotyping.

Value

A `SnpSet` object.

`calls`	Genotype calls (1 - AA, 2 - AB, 3 - BB)
`confs`	Confidence scores 'round(-1000*log2(1-p))'
`SNPQC`	SNP Quality Scores
`batchQC`	Batch Quality Score
`params`	Recalibrated parameters

References

Carvalho B, Bengtsson H, Speed TP, Irizarry RA. Exploration, normalization, and genotype calls of high-density oligonucleotide SNP array data. Biostatistics. 2007 Apr;8(2):485-99. Epub 2006 Dec 22. PMID: 17189563.

Carvalho BS, Louis TA, Irizarry RA. Quantifying uncertainty in genotype calls. Bioinformatics. 2010 Jan 15;26(2):242-9.

Examples

```
## this can be slow
if (require(genomewidesnp6Crlmm) & require(hapmapsnp6)){
  path <- system.file("celFiles", package="hapmapsnp6")

  ## the filenames with full path...
  ## very useful when genotyping samples not in the working directory
  cels <- list.celfiles(path, full.names=TRUE)
  (crlmmOutput <- crlmm(cels))

  ## If gender is known, one should check that the assigned gender is
  ## correct, or pass the integer coding of gender as an argument to the
  ## crlmm function as done below
  gender <- c("female", "female", "male")
  gender[gender == "female"] <- 2
  gender[gender == "male"] <- 1
  ## Not run:   (crlmmOutput <- crlmm(cels, gender=gender))
}

## Not run:
## HPC Example
library(ff)
library(snow)
library(crlmm)
## genotype 50K SNPs at a time
ocProbesets(50000)
## setup cluster - 8 cores on the machine
setCluster(8, "SOCK")

path <- system.file("celFiles", package="hapmapsnp6")
cels <- list.celfiles(path, full.names=TRUE)
```

```
crlmmOutput <- crlmm2(cels)

## End(Not run)
```

`genotype` *Preprocessing and genotyping of Affymetrix arrays.*

Description

Preprocessing and genotyping of Affymetrix arrays.

Usage

```
genotype(filenames, cdfName, batch, mixtureSampleSize = 10^5, eps =0.1,
         verbose = TRUE, seed = 1, sns, probs = rep(1/3, 3),
         DF = 6, SNRMin = 5, recallMin = 10, recallRegMin = 1000,
         gender = NULL, returnParams = TRUE, badSNP = 0.7)
```

Arguments

`filenames`	complete path to CEL files
`cdfName`	annotation package (see also `validCdfNames`)
`batch`	batch variable. See details.
`mixtureSampleSize`	Sample size to be use when fitting the mixture model.
`eps`	Stop criteria.
`verbose`	Logical. Whether to print descriptive messages during processing.
`seed`	Seed to be used when sampling. Useful for reproducibility
`sns`	The sample identifiers. If missing, the default sample names are `basename(filenames)`
`probs`	'numeric' vector with priors for AA, AB and BB.
`DF`	'integer' with number of degrees of freedom to use with t-distribution.
`SNRMin`	'numeric' scalar defining the minimum SNR used to filter out samples.
`recallMin`	Minimum number of samples for recalibration.
`recallRegMin`	Minimum number of SNP's for regression.
`gender`	integer vector (male = 1, female =2) or missing, with same length as filenames. If missing, the gender is predicted.
`returnParams`	'logical'. Return recalibrated parameters from crlmm.
`badSNP`	'numeric'. Threshold to flag as bad SNP (affects batchQC)

Details

For large datasets it is important to utilize the large data support by installing and loading the ff package before calling the `genotype` function. In previous versions of the `crlmm` package, we useed different functions for genotyping depending on whether the ff package is loaded, namely `genotype` and `genotype2`. The `genotype` function now handles both instances.

`genotype` is essentially a wrapper of the `crlmm` function for genotyping. Differences include (1) that the copy number probes (if present) are also quantile-normalized and (2) the class of object returned by this function, `CNSet`, is needed for subsequent copy number estimation. Note that the batch variable that must be passed to this function has no effect on the normalization or genotyping steps. Rather, `batch` is required in order to initialize a `CNSet` container with the appropriate dimensions.

Value

A `SnpSuperSet` instance.

Note

For large datasets, load the 'ff' package prior to genotyping – this will greatly reduce the RAM required for big jobs. See `ldPath` and `ocSamples`.

Author(s)

R. Scharpf

References

Carvalho B, Bengtsson H, Speed TP, Irizarry RA. Exploration, normalization, and genotype calls of high-density oligonucleotide SNP array data. Biostatistics. 2007 Apr;8(2):485-99. Epub 2006 Dec 22. PMID: 17189563.

Carvalho BS, Louis TA, Irizarry RA. Quantifying uncertainty in genotype calls. Bioinformatics. 2010 Jan 15;26(2):242-9.

See Also

`snprma`, `crlmm`, `ocSamples`, `ldOpts`, `batch`, `crlmmCopynumber`

Examples

```
if (require(ff) & require(genomewidesnp6Crlmm) & require(hapmapsnp6)){

  path <- system.file("celFiles", package="hapmapsnp6")
  ## the filenames with full path...
  ## very useful when genotyping samples not in the working directory
  cels <- list.celfiles(path, full.names=TRUE)

  ## Note: one would need at least 10 CEL files for copy number estimation
  ## To use less RAM, specify a smaller argument to ocProbesets
  ocProbesets(50e3)
  batch <- as.factor(rep("A", length(cels)))
  (cnSet <- genotype(cels, cdfName="genomewidesnp6", batch=batch))

  ## when gender is not specified (as in the above example), crlmm tries
  ## to predict the gender from SNPs on chromosome X
```

```
  cnSet$gender

  ## If gender is known, one should check that the assigned gender is
  ## correct. Alternatively, one can pass gender as an argument to the
  ## genotype function.
  gender <- c("female", "female", "male")
  gender[gender == "female"] <- 2
  gender[gender == "male"] <- 1
## Not run:
  cnSet2 <- (cnSet <- genotype(cels, cdfName="genomewidesnp6", batch=batch, gender=as.int

## End(Not run)
  dim(cnSet)
  table(isSnp(cnSet))
}
```

`readIdatFiles` *Reads Idat Files from Infinium II Illumina BeadChips*

Description

Reads intensity information for each bead type from .idat files of Infinium II genotyping BeadChips

Usage

```
readIdatFiles(sampleSheet=NULL, arrayNames=NULL, ids=NULL, path="",
              arrayInfoColNames=list(barcode="SentrixBarcode_A",
                                     position="SentrixPosition_A"),
              highDensity=FALSE, sep="_",
              fileExt=list(green="Grn.idat", red="Red.idat"),
              saveDate=FALSE)
```

Arguments

`sampleSheet`	`data.frame` containing Illumina sample sheet information (for required columns, refer to BeadStudio Genotyping guide - Appendix A).
`arrayNames`	character vector containing names of arrays to be read in. If `NULL`, all arrays that can be found in the specified working directory will be read in.
`ids`	vector containing ids of probes to be read in. If `NULL` all probes found on the first array are read in.
`path`	character string specifying the location of files to be read by the function
`arrayInfoColNames`	(used when `sampleSheet` is specified) list containing elements 'barcode' which indicates column names in the `sampleSheet` which contains the arrayNumber/barcode number and 'position' which indicates the strip number. In older style sample sheets, this information is combined (usually in a column named 'SentrixPosition') and this should be specified as `list(barcode=NULL, position="SentrixPosition")`
`highDensity`	logical (used when `sampleSheet` is specified). If `TRUE`, array extensions '_A', '_B' in sampleSheet are replaced with 'R01C01', 'R01C02' etc.
`sep`	character string specifying separator used in .idat file names.

fileExt	list containing elements 'Green' and 'Red' which specify the .idat file extension for the Cy3 and Cy5 channels.
saveDate	logical. Should the dates from each .idat be saved with sample information?

Details

The summarised Cy3 (G) and Cy5 (R) intensities (on the orginal scale) are read in from the .idat files.

Where available, a sampleSheet data.frame, in the same format as used by BeadStudio (columns 'Sample_ID', 'SentrixBarcode_A' and 'SentrixPosition_A' are required) which keeps track of sample information can be specified.

Thanks to Keith Baggerly who provided the code to read in the binary .idat files.

Value

NChannelSet with intensity data (R, G), and indicator for SNPs with 0 beads (zero) for each bead type.

Author(s)

Matt Ritchie

References

Ritchie ME, Carvalho BS, Hetrick KN, Tavar\'e S, Irizarry RA. R/Bioconductor software for Illumina's Infinium whole-genome genotyping BeadChips. Bioinformatics. 2009 Oct 1;25(19):2621-3.

Examples

```
#RG = readIdatFiles()
```

---

sample.CNSetLM *Dataset of class 'CNSetLM'*

Description

The data for 2119 polymorphic and nonpolymorphic markers on chromosome 1 for the CEPH and Yoruban HapMap samples.

Usage

```
data(sample.CNSetLM)
```

Format

This class has been deprecated. See example below for how to update an existing 'CNSetLM' object to class 'CNSet'.

The data illustrates the CNSetLM-class, with assayData containing the quantile-normalized intensities for the A and B alleles, genotype calls and confidence scores (call and callProbability), and allele-specific copy number (CA and CB). The parameters from the linear model are stored in the lM slot.

Examples

```
## class CNSetLM has been deprecated
data(sample.CNSetLM)
## update to class CNSet
cnSet <- as(sample.CNSetLM, "CNSet")
all(isCurrent(cnSet)) ## is the cnSet object current?
##subsetting
cnSet2 <- cnSet[, 1:5]
stopifnot(batchNames(cnSet2) == "C")
## Not run:
## updating class CNSetLM using ff objects
## a bigger object with multiple batches
if(require(ff)){
outdir <- "/amber1/scratch/rscharpf/jss/hapmap2"
load(file.path(outdir, "container.rda"))
container <- object; rm(object); gc()
container2 <- as(container, "CNSet")
all(isCurrent(container2))
## test replacement methods, subset methods
table(batch(container2))
##generates warning ... would need open, close in the '[' method
invisible(open(nuA(container2)))
xx <- nu(container2, "A")[1:5, ]
nuA(container2)[1:5, ] <- xx
invisible(close(nuA(container2)))
}

## End(Not run)
## ------------------------------------------------
## accessors for the feature-level info
## ------------------------------------------------
chromosome(cnSet)[1:5]
position(cnSet)[1:5]
isSnp(cnSet)[1:5]
## 980 nonpolymorphic markers and 1139 polymoprhic markers
table(isSnp(cnSet))
## ------------------------------------------------
## sample-level statistics computed by crlmm
## ------------------------------------------------
varLabels(cnSet)
## accessors for sample-level statistics
## The signal to noise ratio (SNR)
cnSet$SNR[1:5]
## the skew
cnSet$SKW[1:5]
## the gender (gender is imputed unless specified in the call to crlmm)
table(cnSet$gender)  ## 1=male, 2=female
## ------------------------------------------------
## ------------------------------------------------
##
## accessors for parameters estimated from the linear model for copy
## number (note that the parameters have dimension R x C, where R
## corresponds to the number of features and C corresponds to the
## number of batches)
## ------------------------------------------------ estimate of
## background
```

```
dim(nu(cnSet, "A"))
## background for the A allele in the 2 batches for the
## first 5 markers
nu(cnSet, "A")[1:5, ]
## background for the B allele in the 2 batches for the
## first 5 markers
nu(cnSet, "B")[1:5, ]
## the slope
phi(cnSet, "A")[1:5, ]
## correlation within genotype cluster AA
##corr(cnSet, "AA")[1:5, ]
#### correlation within genotype cluster AB
##corr(cnSet, "AB")[1:5, ]
#### correlation within genotype cluster BB
##corr(cnSet, "BB")[1:5, ]
## ------------------------------------------------

## ------------------------------------------------
## calculating allele-specific copy number
## ------------------------------------------------
## copy number for allele A, first 5 markers, first 2 samples
(ca <- CA(cnSet, i=1:5, j=1:2))
## copy number for allele B, first 5 markers, first 2 samples
(cb <- CB(cnSet, i=1:5, j=1:2))
## total copy number for first 5 markers, first 2 samples
(cn1 <- ca+cb)

## total copy number at first 5 nonpolymorphic loci
index <- which(!isSnp(cnSet))[1:5]
cn2 <- CA(cnSet, i=index, j=1:2)
## note, cb is NA at nonpolymorphic loci
(cb <- CB(cnSet, i=index, j=1:2))
## note, ca+cb will give NAs at nonpolymorphic loci
CA(cnSet, i=index, j=1:2) + cb
## A shortcut for total copy number
cn3 <- totalCopynumber(cnSet, i=1:5, j=1:2)
all.equal(cn3, cn1)
cn4 <- totalCopynumber(cnSet, i=index, j=1:2)
all.equal(cn4, cn2)

## markers 1-5, all samples
cn5 <- totalCopynumber(cnSet, i=1:5)
## all markers, samples 1-5
cn6 <- totalCopynumber(cnSet, j=1:5)

## NOTE: subsetting the object before extracting copy number
##       can be very inefficient when the data set is very large,
##       particularly if using ff objects.  IN particular, subsetting
##       the CNSet object will subset all of the assay data elements
##       and all of the elements in the LinearModelParameter slot
## Not run:
cnsubset <- cnSet[1:5, ]

## End(Not run)
```

snprma *Preprocessing tool for SNP arrays.*

Description

SNPRMA will preprocess SNP chips. The preprocessing consists of quantile normalization to a known target distribution and summarization to the SNP-Allele level.

Usage

```
snprma(filenames, mixtureSampleSize = 10^5, fitMixture = FALSE, eps = 0.1, verbo
snprma2(filenames, mixtureSampleSize = 10^5, fitMixture = FALSE, eps = 0.1, verb
```

Arguments

filenames	'character' vector with file names.
mixtureSampleSize	Sample size to be use when fitting the mixture model.
fitMixture	'logical'. Fit the mixture model?
eps	Stop criteria.
verbose	'logical'.
seed	Seed to be used when sampling.
cdfName	cdfName: 'GenomeWideSnp_5', 'GenomeWideSnp_6'
sns	Sample names.

Details

'snprma2' allows one to genotype very large datasets (via ff package) and also permits the use of clusters or multiple cores (via snow package) to speed up preprocessing.

Value

A	Summarized intensities for Allele A
B	Summarized intensities for Allele B
sns	Sample names
gns	SNP names
SNR	Signal-to-noise ratio
SKW	Skewness
mixtureParams	Parameters from mixture model
cdfName	Name of the CDF

Examples

```
if (require(genomewidesnp6Crlmm) & require(hapmapsnp6) & require(oligoClasses)){
  path <- system.file("celFiles", package="hapmapsnp6")

  ## the filenames with full path...
  ## very useful when genotyping samples not in the working directory
  cels <- list.celfiles(path, full.names=TRUE)
  snprmaOutput <- snprma(cels)
  snprmaOutput[["A"]][1:10,]
  snprmaOutput[["B"]][1:10,]
}
## Not run:
## HPC Example
library(ff)
library(snow)
library(crlmm)
## genotype 50K SNPs at a time
ocProbesets(50000)
## setup cluster - 8 cores on the machine
setCluster(8, "SOCK")

path <- system.file("celFiles", package="hapmapsnp6")
cels <- list.celfiles(path, full.names=TRUE)
snprmaOutput <- snprma2(cels)


## End(Not run)
```

Index