affyPara

April 20, 2011

`bgCorrectPara` *Parallelized Background Correction*

Description

Parallelized functions for background correction of probe intensities.

Usage

```
bgCorrectPara(object,
phenoData = new("AnnotatedDataFrame"), method,
cluster, verbose = getOption("verbose"))
```

Arguments

`object`	An object of class AffyBatch OR a `character` vector with the names of CEL files OR a (partitioned) list of `character` vectors with CEL file names.
`phenoData`	An AnnotatedDataFrame object
`method`	A `character` that defines what background correction method will be used. Available methods are given by `bg.correct.methods`. The name of the method to apply must be double-quoted.
`cluster`	A cluster object obtained from the function makeCluster in the SNOW package. For default `.affyParaInternalEnv$cl` will be used.
`verbose`	A logical value. If `TRUE` it writes out some messages.

Details

`bgCorrectPara` is the parallelized function for background correction of probe intensities. For serial function an more details see bg.correct.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command `makeCluster` generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The `makeXXXcluster` functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Value

An AffyBatch for which the intensities have been background adjusted. For some methods (RMA), only PMs are corrected and the MMs remain the same.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  ##bgc will be the bg corrected version of Dilution
  bgc <- bgCorrectPara(Dilution, method="rma", verbose=TRUE)

  stopCluster()
}

## End(Not run)
```

---

boxplotPara *Parallelized Box Plots for Microarray Data*

Description

Parallelized functions for Box Plots for Microarray Data. And optimized plots for a large number of microarrays.

Usage

```
boxplotPara(object,
nSample=if(length(object)> 200) nSample<-200 else nSample <- length(object),
iqrMethod=TRUE, percent=0.05,
typDef="mean", plot=TRUE,
plotAllBoxes=TRUE,
cluster, verbose = getOption("verbose"))
```

Arguments

nSample	A numeric value. Indicates the number of maximal samples that should be plot at the same boxplot. default: 250
iqrMethod	A logical value, if TRUE the second method will be considered to calculate the "bad" quality Samples otherwise the first Method. See Details. default: TRUE
percent	A numeric value [0.0-1.1]. If iqrMethod=TRUE the second method will be considered to calculate the "bad" quality Samples otherwise the first Method. default: 0.05

typDef	A character value. Indicates how the default Sample should be calculated. As median or mean from all Samples value. Only three possibilities: "media", "mean", c("median", "mean"). default: "mean"
plot	A logical value. Indicates if graphics should be drawn. default: TRUE
plotAllBoxes	A logical value. If TRUE then all Samples will be ploted, if FALSE only bad arrays will be plotted. default: TRUE
object	An object of class AffyBatch OR a character vector with the names of CEL files OR a (partitioned) list of character vectors with CEL file names.
cluster	A cluster object obtained from the function makeCluster in the SNOW package. For default .affyParaInternalEnv$cl will be used.
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")

Details

boxplotPara is the parallelized function for box plots of probe intensities. It is a function to check and control the Data quality of the samples using the boxplot methode. For serial function an more details see boxplot. This function is optimized for huge numbers of microarray data.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command makeCluster generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The makeXXXcluster functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

We need to calculate a default Sample as reference , which has been built from all Samples data. Therefore the first is the calculation of the boxplot.stats, it will be made parallel at the cluster. The calculated values are merged at the master as well as the following calculations, plots and histograms There are two possibility to calculate the limits between the "good-bad" quality Samples: 1. From the differences between defaultSample values (only media, HL nad HU will be considered) and all the samples. At limit will be considered as critical and it help to calculate the "bad" quality Samples. (it is fixed as parameter and thus do it not so sure) 2. From the median and IQR obtained from a boxplot, which is calculated from all Samples values. The outliers of these boxplot are the "bad" quality Samples. It should be as default parameter.

Value

boxplotPara returns a list with elements from the boxplot.stats function ('stats', 'n', 'conf', 'out', 'group', 'names') and QualityPS, values_boxP and results_boxP.

qualityPS	is a list which contains all "bad" quality Arrays classified in levels.
values_boxP	contains the calculated differences and limits, which are used to make the classification of the Arrays in levels.
results_boxP	summary from qualityPS as matrix, which contains only the Arrays that are considered as "bad" quality and in which levels are they classified. Possible values are 0 if the Array is not at this levels and 1 if it is classified as "bad" sample at this level

Author(s)

Esmeralda Vicedo <e.vicedo@gmx.net>, Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.uni-muenchen.de>

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  ##boxplot of Dulution data (affybatch)
  box1 <- boxplotPara(Dilution)

  ## boxplots to a pdf file
  pdf(file="boxplot.pdf", title="AffyBatch Boxplot")
  box2 <- boxplotPara(Dilution)
  dev.off()

  stopCluster()
}

## End(Not run)
```

`vsnInputPara` *Class to contain input data and parameters for parallel vsn functions*

Description

Class extends the class `vsnInput`. The class contains input data and parameters for parallel vsn functions.

Creating Objects

`new("vsnInputPara")`

Slots

`...`: as class `vsnInput`.

`dimAB`**:** The dimension of the complete AffyBatch.

Methods

`dim` Get dimensions of data matrix.

`nrow` Get number of rows of data matrix.

`ncol` Get number of columns of data matrix.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

See Also

`vsn2`

`computeExprSetPara` *Parallel generate a set of expression values*

Description

Parallel generation of a set of expression values from the probe pair information. The set of expression is returned as an ExpressionSet object.

Usage

```
computeExprSetPara(object,
    ids = NULL,
    pmcorrect.method, summary.method,
    summary.param = list(), pmcorrect.param = list(),
    phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
    cluster, verbose = getOption("verbose"))
```

Arguments

`object`	An object of class AffyBatch OR a `character` vector with the names of CEL files OR a (partitioned) list of `character` vectors with CEL file names.
`pmcorrect.method`	The name of the PM adjustement method.
`pmcorrect.param`	A list of parameters for `pmcorrect.method` (if needed/wanted).
`summary.method`	The method used for the computation of expression values
`summary.param`	A list of parameters to be passed to the `summary.method` (if wanted).
`ids`	List of `ids` for summarization.
`phenoData`	An AnnotatedDataFrame object.
`cdfname`	Used to specify the name of an alternative cdf package. If set to `NULL`, the usual cdf package based on Affymetrix' mappings will be used.
`cluster`	A cluster object obtained from the function makeCluster in the SNOW package. For default `.affyParaInternalEnv$cl` will be used.
`verbose`	A logical value. If `TRUE` it writes out some messages.

Details

Parallelized preprocessing function, which goes from raw probe intensities to expression values in one steps: summarization

For the serial function and more details see the function `computeExprSet`.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command `makeCluster` generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The `makeXXXcluster` functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Value

An object of class ExpressionSet.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  esset <- computeExprSetPara(Dilution,
    pmcorrect.method = "pmonly",
    summary.method = "avgdiff",
    verbose = TRUE)

  stopCluster()
}

## End(Not run)
```

---

distributeFiles *Distribute files to slaves*

Description

This function distributes files from the master node to the disk of the slaves in the computer cluster.

Usage

```
distributeFiles(files, to = tempdir(),
protocol = c("R", "RCP", "SCP"), hierarchicallyDist = FALSE,
master=TRUE, delExistTo=FALSE,
full.names=TRUE,
cluster, verbose = getOption("verbose"))
```

Arguments

files	A character vector containing the names of the files.
to	A character that defines the path where the files should be stored at the slaves. Default: tempdir()
protocol	A character that defines the Copy-Protocol: "R", "RCP", "SCP"
hierarchicallyDist	A logical value. If TRUE data will be hierarchically distributed to all slaves. If FALSE at every slave only a part of data is available.
master	A logical value. If TRUE all data will be copied to the 'to' directory at the master node. Default = TRUE

delExistTo	A logical value. If TRUE directory 'to' will be deleted at master and all nodes first. Default = FALSE
full.names	A logical value. If TRUE, the directory path is prepended to the file names (in slot CELfiles). If FALSE, only the file names are returned .
cluster	A cluster object obtained from the function makeCluster in the SNOW package. For default .affyParaInternalEnv$cl will be used.
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")

Details

This function distributes files from the master node to the disk of the slaves in the computer cluster. First the vector of files get partitioned by the number of slaves. Then the parts will be copied to the to directory at the slaves. If hierarchicallyDist is TRUE, all slaves change the files among each other and in the end at every slave all files are located. (But this is not necessary for distributed computing with the affyPara package.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command makeCluster generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The makeXXXcluster functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Value

A list of two objects

to	A character that defines the path where the files are located at the slaves.
CELfiles	A list of characters, how the files are distributed to the slaves. Depending on full.names only the filenames or path/filenames.

Warning

For protocol "R" hierarchically distribution not yet available.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)

makeCluster(10)

path <- "tmp/CELfiles"
CELfiles <- list.files(path,full.names=TRUE)

distList <- distributeFiles(CELfiles, protocol="RCP", verbose=TRUE)

stopCluster()

## End(Not run)
```

MAplotPara *Parallelized relative M vs. A plots for Microarray Data*

Description

Parallelized creation of M vs A plots. Where M is determined relative to a specified chip or to a pseudo-median reference chip. And optimized plots for a large number of microarrays.

Usage

```
MAplotPara(object,
                        log=TRUE,
                        type=c("both","pm","mm"),
                        ref=NULL,
                        which=NULL,
                        ref.title="vs mean ref.Array",
                        subset=NULL,
                        span=1/4,
                        show.statistics=TRUE,
                        family.loess ="gaussian",
                        pch=".",
                        plot =TRUE,
                        cutoff =0.5,# add parameter to generic function ma.plot
                        level=1,
                        cluster, verbose = getOption("verbose"),
                        ... )
```

Arguments

object	An object of class AffyBatch OR a character vector with the names of CEL files OR a (partitioned) list of character vectors with CEL file names.
...	Additional parameters for the routine.
log	A logical value. TRUE as default. Computes logarithms of Array's intensities.
type	Defines how the Affimetrix Array's intensities should be considered (only for Affymetrix: perfect match(pm), mismatch or both).
ref	gives the possibility to define some array of the Affybatch object as 'medianchip' otherwise it will be calculate as reference median array.
which	A list of index samples from the object class, which indicates the samples to be ploted.
ref.title	character- gives a 'title' to label the plots by ma.plot function.
subset	A set of indices to use when drawing the loess curve.
span	span to be used for loess fit.
family.loess	"guassian" or "symmetric" as in loess.
show.statistics	A logical value. TRUE as default. If true some summary statistics of the M values are drawn.

pch	Graphical plotting 'character'. Default Value "." equivalently to pch = 46 from the function points
plot	A logical value. TRUE as default, MAplots will be drawn otherwise only the "bad" quality arrays will be calculated.
cutoff	numerical between [0.0-1.0]. As default 0.5. It is considered as limit for the sigma parameter, which is one of the three classificators for the 'bad' quality arrays by MAplotsPara.
level	level- numerical - indicates which level of "bad" quality arrays should be plot if plotDraw =TRUE: 1 - only first level "bad" quality will be considered. First level "bad" array quality are the arrays considered as "bad" after the three possible parameter: S, loess, and sigma 2 - first level "bad" quality and second level will be considered. Second level "bad" quality Arrays are the arrays which has been classified as bad after two of the three possible parameter 3 - all levels will be plot : first, second and third. Third level "bad" quality Arrays are the arrays which are considered as "bad" after one of the three parameter.
cluster	A cluster object obtained from the function makeCluster in the SNOW package. For default .affyParaInternalEnv$cl will be used.
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")

Details

MAplotPara is a function based on the generic function ma.plot from affy package. Only the following parameters are original for MAplotPAra: cluster, object, cuttof, plot, level The parameter ref.fn=c("median","mean") is not allowed because it is not possible to calculate the reference median array as parallelized.

MAplotPara is the parallelized function for MA plots of probe intensities. It is a function to check and control the Data quality of the samples using the MA plot method. For serial function an more details see boxplot. This function is optimized for huge numbers of microarray data.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command makeCluster generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The makeXXXcluster functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Value

MAplotPara return a list with four elements: values_MAP, loess_y, quality_MAP, results_MAP.

values_MAP	is a matrix which contains the calculated values for all arrays of the object (sampleNames, S, osc_Loess, sigma, var_sigma).
loess_y	contains the y values (50 points) of the loess curve, which are used to calculate the osc_Loess from values_MAP
quality_MAP	list which contains all "bad" quality Arrays
results_MAP	summary from quality_MAP as matrix, which contains only the Arrays that are considered as "bad" quality and in which levels are they classified. Possible values are 0 if the Array is not at this levels and 1 if it is classified as "bad" sample at this level.

Author(s)

Esmeralda Vicedo <e.vicedo@gmx.net>, Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.uni-muenchen.de>

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  ##MA of Dilution data (affybatch)
  ma1 <- MAplotPara(Dilution)

  ## MAplot to a pdf file
  pdf(file="maplot.pdf", title="AffyBatch MAplot")
  ma2 <- MAplotPara(Dilution)
  dev.off()

  stopCluster()
}

## End(Not run)
```

---

mergeSplitObjects *Merge a list of split objects*

Description

Functions to merge or combine a list of split objects (AffyBatch, Matrix).

Usage

```
mergeAffyBatches(abatch.list, description = NULL, notes = character(0))
combineMatrices(matrix.list, verbose = getOption("verbose"))
```

Arguments

abatch.list	A list of objects of class AffyBatch.
description	A MIAME object.
notes	A character vector of explanatory text.
matrix.list	A list of objects of class matrix.
verbose	A logical value. If TRUE it writes out some messages.

Details

Functions to merge or combine a list of split objects.

mergeAffyBatches Merges a list of AffyBatches to one AffyBatch.

combineMatrices Combines a list of matrices by columns to one matrix.

Value

`mergeAffyBatches`
: Returns ONE object of class AffyBatch.

`combineMatrices`
: Returns ONE object of class matrix.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  #split AffyBatch
  abatch.list<- splitAffyBatch(Dilution, 2)

  #Merge AffyBatch
  AffyBatch <- mergeAffyBatches(abatch.list)

  # Create matrices
  a <- matrix(1:25, nrow=5)
  b <- matrix(101:125, nrow=5)
  matrix.list <- list(a,b)

  # Combine matrices
  combineMatrices(matrix.list)
}
```

---

normalizeAffyBatchConstantPara

Parallelized scaling normalization

Description

Parallelized scaling normalization of arrays.

Usage

```
normalizeAffyBatchConstantPara(object,
      refindex = 1, FUN = mean, na.rm = TRUE,
      phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
      cluster, verbose = getOption("verbose"))
```

Arguments

`object`	An object of class AffyBatch OR a `character` vector with the names of CEL files OR a (partitioned) list of `character` vectors with CEL file names.
`refindex`	The index of the array used as reference.

FUN	A function generating a value from the intensities on an array. Typically mean or median.
na.rm	Paramater passed to the function FUN. A logical value indicating whether NA values should be stripped before the computation proceeds.
phenoData	An AnnotatedDataFrame object.
cdfname	Used to specify the name of an alternative cdf package. If set to NULL, the usual cdf package based on Affymetrix' mappings will be used.
cluster	A cluster object obtained from the function makeCluster in the SNOW package. For default .affyParaInternalEnv$cl will be used.
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")

Details

Parallelized scaling normalization of arrays. This means that all the array are scaled so that they have the same mean value.

For the serial function and more details see the function normalize.constant.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command makeCluster generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The makeXXXcluster functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Value

An AffyBatch of normalized objects.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  AffyBatch <- normalizeAffyBatchConstantPara(Dilution, verbose=TRUE)

  stopCluster()
}

## End(Not run)
```

`normalizeAffyBatchInvariantsetPara` *Parallelized Invariante Set normalization*

Description

Parallelized normalization of arrays using an invariant set.

Usage

```
normalizeAffyBatchInvariantsetPara(object,
    prd.td = c(0.003, 0.007), baseline.type = c("mean", "median", "pseudo-mean",
    type = c("separate", "pmonly", "mmonly", "together"),
    phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
    cluster, verbose = getOption("verbose"))
```

Arguments

`object`	An object of class AffyBatch OR a `character` vector with the names of CEL files OR a (partitioned) list of `character` vectors with CEL file names.
`prd.td`	A cutoff parameter for normalization.
`baseline.type`	Specify how to determine the baseline array (mean, median).
`type`	A string specifying how the normalization should be applied.
`phenoData`	A AnnotatedDataFrame object.
`cdfname`	Used to specify the name of an alternative cdf package. If set to `NULL`, the usual cdf package based on Affymetrix' mappings will be used.
`cluster`	A cluster object obtained from the function makeCluster in the SNOW package. For default `.affyParaInternalEnv$cl` will be used.
`verbose`	A logical value. If `TRUE` it writes out some messages. default: getOption("verbose")

Details

Parallelized normalization of arrays using an invariant set. The set of invariant intensities between data and ref is found through an iterative process (based on the respective ranks the intensities). This set of intensities is used to generate a normalization curve by smoothing.

For the serial function and more details see the function `normalize.invariantset`.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command `makeCluster` generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The `makeXXXcluster` functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Value

An AffyBatch of normalized objects.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  AffyBatch <- normalizeAffyBatchInvariantsetPara(Dilution, verbose=TRUE)

  stopCluster()
}

## End(Not run)
```

---

normalizeAffyBatchLoessIterPara

Parallelized partial loess normalization with permutation

Description

Parallelized partial cyclic loess normalization of arrays with permutation.

Usage

```
normalizeAffyBatchLoessIterPara(object,
percentPerm = 0.75,
phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
type=c("separate","pmonly","mmonly","together"),
subset = NULL,
epsilon = 10^-2, maxit = 1, log.it = TRUE,
span = 2/3, family.loess ="symmetric",
cluster, verbose = getOption("verbose"))
```

Arguments

object	An object of class AffyBatch OR a character vector with the names of CEL files OR a (partitioned) list of character vectors with CEL file names.
percentPerm	Percent of permutations to do.
phenoData	An AnnotatedDataFrame object.
cdfname	Used to specify the name of an alternative cdf package. If set to NULL, the usual cdf package based on Affymetrix' mappings will be used.
type	A string specifying how the normalization should be applied.
subset	a subset of the data to fit a loess to.
epsilon	a tolerance value (supposed to be a small value - used as a stopping criterium).
maxit	maximum number of iterations.

log.it	logical. If TRUE it takes the log2 of mat
span	parameter to be passed the function loess
family.loess	parameter to be passed the function loess. "gaussian" or "symmetric" are acceptable values for this parameter.
cluster	A cluster object obtained from the function makeCluster in the SNOW package. For default .affyParaInternalEnv$cl will be used.
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")

Details

Parallelized partial cyclic loess normalization of arrays with permutation. This is a new kind of normalization based on cyclic loess normalization.

In the partial cyclic loess normalization the loess normalization will be done only at the slaves with the arrays at the slaves. Therefore we only have to do loess normalization for some pairs and have a big saving of time. But this is no enough for good normalization. We have to do some iterations of array permutation between the slaves and again loess normalization at the slaves. If we did about 75 percent of the complete cyclic loess normalization we can achieve same results and save computation time.

For the similar serial function and more details to loess normalization see the function normalize.AffyBatch.loes

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command makeCluster generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The makeXXXcluster functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

In the loess normalization the arrays will compared by pairs. Therefore at every node minimum two arrays have to be!

Value

An AffyBatch of normalized objects.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  AffyBatch <- normalizeAffyBatchLoessIterPara(percentPerm=0.75, Dilution, verbose=TRUE)

  stopCluster()
}

## End(Not run)
```

`normalizeAffyBatchLoessPara` *Parallelized loess normalization*

Description

Parallelized loess normalization of arrays.

Usage

```
normalizeAffyBatchLoessPara(object,
phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
type=c("separate","pmonly","mmonly","together"),
subset = NULL,
epsilon = 10^-2, maxit = 1, log.it = TRUE,
span = 2/3, family.loess ="symmetric",
cluster, verbose = getOption("verbose"))
```

Arguments

`object`	An object of class AffyBatch OR a `character` vector with the names of CEL files OR a (partitioned) list of `character` vectors with CEL file names.
`phenoData`	An AnnotatedDataFrame object.
`cdfname`	Used to specify the name of an alternative cdf package. If set to `NULL`, the usual cdf package based on Affymetrix' mappings will be used.
`type`	A string specifying how the normalization should be applied.
`subset`	a subset of the data to fit a loess to.
`epsilon`	a tolerance value (supposed to be a small value - used as a stopping criterium).
`maxit`	maximum number of iterations.
`log.it`	logical. If `TRUE` it takes the log2 of mat
`span`	parameter to be passed the function loess
`family.loess`	parameter to be passed the function loess. "gaussian" or "symmetric" are acceptable values for this parameter.
`cluster`	A cluster object obtained from the function makeCluster in the SNOW package. For default `.affyParaInternalEnv$cl` will be used.
`verbose`	A logical value. If `TRUE` it writes out some messages. default: getOption("verbose")

Details

Parallelized loess normalization of arrays.

For the serial function and more details see the function `normalize.AffyBatch.loess`.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command `makeCluster` generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object

handling is required. The `makeXXXcluster` functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

In the loess normalization the arrays will compared by pairs. Therefore at every node minimum two arrays have to be!

Value

An AffyBatch of normalized objects.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  AffyBatch <- normalizeAffyBatchLoessPara(Dilution, verbose=TRUE)

  stopCluster()
}

## End(Not run)
```

---

normalizeAffyBatchQuantilesPara

Parallelized quantile normalization

Description

Parallelized normalization of arrays based upon quantiles.

Usage

```
normalizeAffyBatchQuantilesPara(object,
    phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
    type = c("separate", "pmonly", "mmonly", "together"),
    cluster, verbose = getOption("verbose"))

normalizeQuantilesPara(cluster, type, object.length, verbose = getOption("verbos
```

Arguments

`object`	An object of class AffyBatch OR a `character` vector with the names of CEL files OR a (partitioned) list of `character` vectors with CEL file names.
`phenoData`	An AnnotatedDataFrame object.

cdfname	Used to specify the name of an alternative cdf package. If set to NULL, the usual cdf package based on Affymetrix' mappings will be used.
type	A string specifying how the normalization should be applied.
cluster	A cluster object obtained from the function makeCluster in the SNOW package. For default .affyParaInternalEnv$cl will be used.
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")
object.length	Number of samples, which should be normalized.

Details

Parallelized normalization of arrays based upon quantiles. This method is based upon the concept of a quantile-quantile plot extended to n dimensions. No special allowances are made for outliers.

For the serial function and more details see the function normalize.AffyBatch.quantiles.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command makeCluster generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The makeXXXcluster functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

normalizeQuantilesPara is a internal function which will be executed at all slaves.

normalizeQuantilesPara Function for quantile normalization.

Value

An AffyBatch of normalized objects.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  AffyBatch <- normalizeAffyBatchQuantilesPara(Dilution, verbose=TRUE)

  stopCluster()
}

## End(Not run)
```

preproPara *Parallelized preprocessing*

Description

Parallelized preprocessing function, which goes from raw probe intensities to expression values in three steps: Background correction, normalization and summarization

Usage

```
preproPara(object,
    bgcorrect = TRUE, bgcorrect.method = NULL, bgcorrect.param = list(),
    normalize = TRUE, normalize.method = NULL, normalize.param = list(),
    pmcorrect.method = NULL, pmcorrect.param = list(),
    summary.method = NULL, summary.param = list(),
    ids = NULL,phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
    cluster, verbose = getOption("verbose"))
```

Arguments

object	An object of class AffyBatch OR a character vector with the names of CEL files OR a (partitioned) list of character vectors with CEL file names.
bgcorrect	A boolean to express whether background correction is wanted or not.
bgcorrect.method	The name of the background adjustment method to use.
bgcorrect.param	A list of parameters for bgcorrect.method (if needed/wanted)
normalize	A boolean to express whether normalization is wanted or not.
normalize.method	The name of the normalization method to use.
normalize.param	A list of parameters to be passed to the normalization method (if wanted).
pmcorrect.method	The name of the PM adjustment method.
pmcorrect.param	A list of parameters for pmcorrect.method (if needed/wanted).
summary.method	The method used for the computation of expression values.
summary.param	A list of parameters to be passed to the summary.method (if wanted).
ids	List of ids for summarization
phenoData	An AnnotatedDataFrame object.
cdfname	Used to specify the name of an alternative cdf package. If set to NULL, the usual cdf package based on Affymetrix' mappings will be used.
cluster	A cluster object obtained from the function makeCluster in the SNOW package. For default .affyParaInternalEnv$cl will be used.
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")

Details

Parallelized preprocessing function, which goes from raw probe intensities to expression values in three steps: Background correction, normalization and summarization

For the serial function and more details see the function `expresso`.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command `makeCluster` generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The `makeXXXcluster` functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Available methods:

`bgcorrect.method`**:** see `bgcorrect.methods()`

`normalize.method`**:** 'quantil', 'constant', 'invariantset','loess'

`summary.method`**:** see `generateExprSet.methods()` and 'none'.

Value

An object of class ExpressionSet.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  esset <- preproPara(Dilution,
    bgcorrect = TRUE, bgcorrect.method = "rma",
    normalize = TRUE, normalize.method = "quantiles",
    pmcorrect.method = "pmonly",
    summary.method = "avgdiff",
    verbose = TRUE)

  stopCluster()
}

## End(Not run)
```

qa *Parallel Quality Assessment Summary*

Description

Creates a Summary Matrix from parallel quality assessment results.

Usage

```
summaryM1M2Para(method1, method2,
level, verbose=FALSE)
```

Arguments

method1	Result object form boxplotPara.
method2	Result object from MAplotPara.
level	level- numerical - indicates which level of "bad" quality arrays should be plot if plotDraw =TRUE: 1 - only first level "bad" quality will be considered. First level "bad" array quality are the arrays considered as "bad" after the three possible parameter: S, loess, and sigma 2 - first level "bad" quality and second level will be considered. Second level "bad" quality Arrays are the arrays which has been classified as bad after two of the three possible parameter 3 - all levels will be plot : first, second and third. Third level "bad" quality Arrays are the arrays which are considered as "bad" after one of the three parameter.
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")

Details

summaryM1M2Para creates a Summary Matrix from parallel quality assessment results. In the rows there are the arrays and in the colums the qa-methods: 0 = good quality, 1 = bad quality.

If the rowSum is bigger than 2, than the arrays should be considered as bad quality.

Value

A matrix of all arrays (rows) and qa-methods (colums): 0 = good quality, 1 = bad quality

Author(s)

Esmeralda Vicedo <e.vicedo@gmx.net>, Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.uni-muenchen.de>

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3, type='MPI')

  box1 <- boxplotPara(Dilution)
```

```
  ma1 <- MAplotPara(Dilution)

  summaryM1M2Para(box1, ma1, level=3)

  stopCluster()
}

## End(Not run)
```

`readAffybatchPara` *Parallelized Read-AffyBatch function*

Description

Parallelization of the read.affybatch function. This parallel implementation is especially useful for multicore machines.

Usage

```
read.affybatchPara(object,
phenoData = new("AnnotatedDataFrame"),
description = NULL, notes = "",
cluster, verbose=getOption("verbose"))
```

Arguments

`object`	An object of a `character` vector with the names of CEL files OR a (partitioned) list of `character` vectors with CEL file names.
`phenoData`	An AnnotatedDataFrame object.
`description`	a 'MIAME' object.
`notes`	notes.
`cluster`	A cluster object obtained from the function makeCluster in the SNOW package. For default `.affyParaInternalEnv$cl` will be used.
`verbose`	A logical value. If `TRUE` it writes out some messages. default: getOption("verbose")

Details

Parallelized creation of an AffyBatch object. Especially useful on multi-core machines to accelerate the creation of the AffyBatch object.

For the serial function and more details see the function `read.affybatch`.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command `makeCluster` generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The `makeXXXcluster` functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Value

An AffyBatch object.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
celpath <- system.file("celfiles", package="affydata")
    fns <- list.celfiles(path=celpath,full.names=TRUE)

makeCluster(3)

##read a text celfile
    abatch <- read.affybatchPara(fns[2], verbose=TRUE)

stopCluster()
}

## End(Not run)
```

---

removeDistributedFiles

Remove distributed files from slaves

Description

This function removes distributed files from a special path at the disk at all slaves in a computer cluster.

Usage

```
removeDistributedFiles(path=tempdir(), cluster, master=TRUE, verbose = getOption
```

Arguments

path	A character that defines which path (inclusive files) should be removed at every slave. Default: tempdir()
cluster	A cluster object obtained from the function makeCluster in the SNOW package. For default .affyParaInternalEnv$cl will be used.
master	A logical value. If TRUE the files will be removed from the master. default: TRUE
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")

Details

This function removes distributed files from a special path at the disk at all slaves in a computer cluster.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command makeCluster generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in

the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The `makeXXXcluster` functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Value

If `verbose = TRUE`, result of removing (successfully / not successfully) will be noticed with a message.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)

makeCluster(10)

removeDistributedFiles(verbose=TRUE)

stopCluster()

## End(Not run)
```

---

rmaPara *Parallelized PMA preprocessing*

Description

Parallelized preprocessing function, which converts an AffyBatch into an ExpressionSet using the robust multi-array average (RMA) expression measure.

Usage

```
rmaPara(object,
ids = NULL,
phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
cluster, verbose = getOption("verbose"), summary.method="medianpolish")
```

Arguments

`object`	An object of class AffyBatch OR a `character` vector with the names of CEL files OR a (partitioned) list of `character` vectors with CEL file names.
`ids`	List of `ids` for summarization
`phenoData`	An AnnotatedDataFrame object.
`cdfname`	Used to specify the name of an alternative cdf package. If set to `NULL`, the usual cdf package based on Affymetrix' mappings will be used.
`cluster`	A cluster object obtained from the function makeCluster in the SNOW package. For default `.affyParaInternalEnv$cl` will be used.

verbose A logical value. If TRUE it writes out some messages. default: getOption("verbose")

summary.method
The method used for the computation of expression values

Details

Parallelized preprocessing function, which goes from raw probe intensities to expression values using the robust multi-array average (RMA) expression measure: Background correction: rma; Normalization: quantile; Summarization: medianpolish

For the serial function and more details see the function rma.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command makeCluster generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The makeXXXcluster functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

This is a wrapper function for preproPara.

Value

An object of class ExpressionSet.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  esset <- rmaPara(Dilution)

  stopCluster()
}

## End(Not run)
```

---

snow-startstop *Starting and Stopping SNOW Clusters*

Description

Functions to start and stop a SNOW cluster and to set default cluster options. Wrapper around original start-stop commands from SNOW to hide the cluster object.

Usage

```
makeCluster( ...)
stopCluster(cl)
```

Arguments

`...`	cluster option specifications, for details see SNOW package.
`cl`	cluster object.

Details

`makeCluster` starts a cluster of the specified or default type and returns NO reference to the cluster. The reference is stored in an internal environment (.affyParaInternalEnv) and will be automatically used from the functions in affyPara. For further parameters and documentation see the SNOW package.

`stopCluster` should be called to properly shut down the cluster before exiting R. If it is not called it may be necessary to use external means to ensure that all slave processes are shut down.

Examples

```
  ## Not run:
makeCluster(2)
stopCluster()

## End(Not run)
```

`splitObjects` *Functions to split objects into parts*

Description

Functions to split an AffyBatch, a `list` of files and a `matrix` into several objects for distributed computing. If possible objects will be of the same size.

Usage

```
splitAffyBatch(abatch, number.part)
splitFileVector(fileVec, number.part)
splitMatrix(matrix, number.part)
```

Arguments

`abatch`	An object of class AffyBatch.
`fileVec`	A `character` vector containing the names of the files.
`matrix`	An object of class matrix.
`number.part`	Number of parts to split the object

Details

splitAffyBatch Splits an AffyBatch into a list of AffyBatches.

splitFileVector Splits a character vector of file names into a list of character vectors with file names.

splitMatrix Splits a matrix by columns into a list of matrices.

These functions use the functions splitIndices and splitCols from the SNOW package.

Value

A list of the split objects.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  spAffyB <- splitAffyBatch(Dilution, 2)
}
```

---

vsnPara *Parallel fir of the vsn model*

Description

These parallel functions fit the vsn model to intensity data in an AffyBatch. They hav the same functionality than the vsn methods in the vsn package but are implemented in parallel (and only supports an AffyBatch as input data).

Usage

```
vsn2Para(object,
phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
reference, subsample,
...,
cluster, verbose = getOption("verbose"))

justvsnPara(object,
...,
cluster, verbose = getOption("verbose"))

vsnrmaPara(object,
pmcorrect.method="pmonly", pmcorrect.param=list(),
summary.method="medianpolish", summary.param=list(),
ids=NULL,
phenoData = new("AnnotatedDataFrame"), cdfname = NULL,
...,
cluster, verbose = getOption("verbose"))
```

Arguments

object	An object of class AffyBatch OR a character vector with the names of CEL files OR a (partitioned) list of character vectors with CEL file names.
phenoData	An AnnotatedDataFrame object.
cdfname	Used to specify the name of an alternative cdf package. If set to NULL, the usual cdf package based on Affymetrix' mappings will be used.
subsample	Integer of length 1. If specified, the model parameters are estimated from a subsample of the data of size subsample only, yet the fitted transformation is then applied to all data. For large datasets, this can substantially reduce the CPU time and memory consumption at a negligible loss of precision.
reference	Optional, a vsn object from a previous fit. If this argument is specified, the data are normalized "towards" an existing set of reference arrays whose parameters are stored in the object reference. If this argument is not specified, then the data are normalized "among themselves".
...	Further arguments that get passed and are similar to vsn2.
cluster	A cluster object obtained from the function makeCluster in the SNOW package. For default .affyParaInternalEnv$cl will be used.
verbose	A logical value. If TRUE it writes out some messages. default: getOption("verbose")
pmcorrect.method	The name of the PM adjustement method.
pmcorrect.param	A list of parameters for pmcorrect.method (if needed/wanted).
summary.method	The method used for the computation of expression values
summary.param	A list of parameters to be passed to the summary.method (if wanted).
ids	List of ids for summarization

Details

For the serial function and more details see the function vsn2.

For using this function a computer cluster using the SNOW package has to be started. Starting the cluster with the command makeCluster generates an cluster object in the affyPara environment (.affyParaInternalEnv) and no cluster object in the global environment. The cluster object in the affyPara environment will be used as default cluster object, therefore no more cluster object handling is required. The makeXXXcluster functions from the package SNOW can be used to create an cluster object in the global environment and to use it for the preprocessing functions.

Value

An AffyBatch of normalized objects.

Author(s)

Markus Schmidberger <schmidb@ibe.med.uni-muenchen.de>, Ulrich Mansmann <mansmann@ibe.med.u

Examples

```
## Not run:
library(affyPara)
if (require(affydata)) {
  data(Dilution)

  makeCluster(3)

  AB1 <- justvsnPara(Dilution, verbose=verbose )

  stopCluster()
}

## End(Not run)
```

Index