Agi4x44PreProcess

November 11, 2009

R topics documented:

`Agi4x44PreProcess-package`
PreProcessing of Agilent 4x44 array data

Description

Agi4x44PreProcess Package Overview

Details

The package allows the preprocessing of Agilent 4x44 array data produced by the Agilent Feature Extraction (AFE) image analysis software. The AFE extracts foreground and background signals, as well as some quality flags. All the extracted information is assembled into the componenents of a 'RGList' object (see 'limma' package)

The preprocessing includes: background correction, normalization and filtering probes according to different quality flags that are produced by the AFE.

A 'target' file and the corresponding data files produced by the AFE image analysis software are required as inputs.

The preprocessing steps are the following: - reading the targets file - reading the array data samples obtained with AFE - Background correction - Normalization between samples - Filtering probes by their Quality Flag - Summarizing replicated probes - Creating and ExpressionSet object with the processed data

The package also contains two specific functions that allow the users to explore the architecture of the chip in terms of probe replication and gene replication. In the first case, it identifies non-control replicated probes (Probe Sets) that are spread over the chip with the propouse of evaluating its reproducibility. In the second case, it picks those genes (according to the ACCNUM code obtained from the corresponding Bioconductor annotation package) that are interrogated by different probes in different locations. These groups of genes are termed 'Gene Sets' .

The package also contains standard graphical microarray utilities that allow the users to evaluate the quality of the data. These graphics also allow to make a decision about what sort of foreground and background signals, among those provided by the AFE, are going to be used in the analysis. A graphical inspection of the data also might help to dedice what background signal correction and normalization between samples could be more suitable to perform.

There are also utility functions that write files across different stages of the processing protocol. These files include the probes list, with information such as their quality flag, normalized intensity and the corresponding information obtained from its annotation package.

Author(s)

Pedro Lopez-Romero ⟨plopez@cnic.es⟩

References

Agilent Feature Extraction Reference Guide http://www.Agilent.com

Gordon K. Smyth, M. Ritchie, N. Thorne, J. Wettenhall (2007). limma: Linear Models for Microarray Data User's Guide.

Bolstad, B. M. (2001), Probe level quantile normalization of high density oligonucleotide array data. Unpublished Manuscript: http://bmbolstad.com/stuff/qnorm.pdf

Bolstad, B. M., Irizarry R. A., Astrand, M., and Speed, T. P. (2003), A comparison of normalization methods for high density oligonucleotide array data based on bias and variance. Bioinformatics 19, 185-193.

Smyth, G. K. (2005). Limma: linear models for microarray data. In: 'Bioinformatics and Computational Biology Solutions Using R and Bioconductor'. R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, W. Huber (eds), Springer, New York, pages 397 - 420

Examples

```
        ## Not run:  reading target file and Agilent Feature Extraction data files

               targets=read.targets(infile="targets.txt")
               dd=read.AgilentFE(targets,makePLOT=TRUE)
        ## End(Not run)

        ## Not run:
        data(dd)
        data(targets)

## End(Not run)
        ## Not run: Non-Control replicated Probes

        ## Not run:
        CV.rep.probes(dd,"hgug4112a.db",
                foreground="MeanSignal",raw.data=TRUE,writeR=TRUE,targets)

## End(Not run)
        ## Not run: genes replicated - ensembl

        ## Not run:
        genes.rpt.agi(dd,annotation.package="hgug4112a.db",raw.data=TRUE,
                WRITE.html=TRUE,REPORT=TRUE)

## End(Not run)
        ## Not run: NORMALIZATION (here the foreground and background are chosen)

        ## Not run:
        ddNORM=BGandNorm(dd,BGmethod='half',NORMmethod='quantile',
                        foreground='MeanSignal',background='BGMedianSignal',
                        offset=50,makePLOTpre=TRUE,makePLOTpost=TRUE)

## End(Not run)
        ## Not run: FILTERING PROBES

        ## Not run:
        ddFILT=filter.probes(ddNORM,
                control=TRUE,
```

```
                wellaboveBG=TRUE,
                isfound=TRUE,
                wellaboveNEG=TRUE,
                sat=TRUE,
                PopnOL=TRUE,
                NonUnifOL=TRUE,
                nas=TRUE,
                limWellAbove=75,
                limISF=75,
                limNEG=75,
                limSAT=75,
                limPopnOL=75,
                limNonUnifOL=75,
                limNAS=100,
                makePLOT=TRUE,annotation.package="hgug4112a.db",flag.counts=TRUE,targets)

## End(Not run)
        ## Not run: SUMMARIZING PROBES

        ## Not run:
        ddPROC=summarize.probe(ddFILT,makePLOT=TRUE,targets)

## End(Not run)
        ## Not run: CREATING EXPRESIONSET OBJECT

        ## Not run:
        esetPROC=build.eset(ddPROC,targets,makePLOT=TRUE,
                annotation.package="hgug4112a.db")
        dim(esetPROC)

        ## End(Not run)
        ## Not run: WRITING EXPRESIONSET OBJECT: ProcessedData.txt

        ## Not run:
        write.eset(esetPROC,ddPROC,"hgug4112a.db",targets)

## End(Not run)

        ## Not run:  MAPPING VARIABLE
        ## Not run:
        mappings=build.mappings(esetPROC,annotation.package="hgug4112a.db")
        names(mappings)

        ## End(Not run)

        ## Not run: Gene Set Enrichment Analysis at: http://www.broad.mit.edu/gsea

        ## Not run:
        gsea.files(esetPROC,targets,annotation.package="hgug4112a.db")

## End(Not run)
```

BGandNorm *Background Correction and Normalization Between Arrays*

Description

For Background correction it uses the 'backgroundCorrect' function of 'limma' package ('half','normexp'). For Normalization between arrays it uses 'limma' function 'normalizeBetweenArrays' ('quantile','vsn').

Usage

```
BGandNorm(RGlist, BGmethod, NORMmethod, foreground,
background, offset, makePLOTpre, makePLOTpost)
```

Arguments

RGlist	an 'RGList' object
BGmethod	Method for the BG corection. Possible values are: 'none','half','normexp'. See ?backgroundCorrect in limma package for details
NORMmethod	Method for Norm between arrays. Possible values can be: 'none','quantile',vsn'. See ?normalizeBetweenArrays in limma package
foreground	Foreground Signal to be used for the analysis. Possible values are 'MeanSignal','ProcessedSignal'
background	Background Signal to be used for the BG correction. The values can be: 'BGMedianSignal','BGUsed'
offset	numeric value to add to the intensities before log transforming. The offset shrunks the log ratios towards zero at the lower intensities. See limma user guide for details
makePLOTpre	density Plots, box plots, MVA plots and RLE plots with the raw signal
makePLOTpost	density Plots, box plots, MVA plots and RLE plots with the normalized signal

Details

In order to make direct comparisons of data coming from different chips it is important to remove sources of variation of non biological nature that may exists between arrays. Systematic non-biological differences between chips become relevant in several obvious ways especially during labeling and hybridization, and bias the relative measures on any two chips when we want to quantify differences due to different treatment between two samples. Normalization is the attempt to compensate for systematic technical differences between chips, to see more clearly the systematic biological differences between samples. First data are background corrected. We produce a Background Subtracted Signal. The Background Signal Used depends on the AFE settings for the type of background method calculation and the settings for spatial detrend. Usually, the Background Signal Used is the sum of the Local Background Signal + the Spatial Detrending Surface Value computed by the AFE software. For the Background correction we use the 'backgroundCorrect' function of 'limma' package with options <'half','normexp'> This function is designed to produce positive corrected intensities. First, any intensity value lower than 0.5 is reset to be equal to 0.5. Besides, and offset value (normally 50) is used. This offset value adds a constant to the intensity values before log-transforming, so that the log ratios are shrunk towards zero at the lower intensities. After background correction, data are normalized between arrays using 'limma' function 'normalizeBetweenArrays' with options <'quantile','vsn'>

For foreground signal,the user can choose between the 'MeanSignal' and the 'ProcessedSignal' and between the 'BGMedianSignal' and the 'BGUsed' for background correction. The user may want to have a look at different graphics (density plots, etc ...) in order to decide what signal is more suitable to use. For details about signal processing see AFE User Guide. 'MeanSignal' is the spot Raw mean signal. 'ProcessedSignal' is the signal processed by the Agilent Feature Extraction image

analysis software (AFE). It contains the Multiplicatively Detrend Bacground Substracted Signal if the detrending is selected and it helps. If the detrending does not help, the 'ProcessedSignal' will be the Bacground Subtracted Signal. 'BGMedianSignal' is the Median local background signal. 'BGUsed' depends on the AFE software settings for the type of background method calculation and the setting for the spatial detrend. Usually, the Background Signal Used is the sum of the local bacground + the spatial detrending surface value computed by the AFE software. To view the values used to calculate this variable using different bakground signals and settings of spatial detrend and global background adjust, see Table 33 on page 213 of the AFE User Guide. Limma function 'backgroundCorrect' is used for the BG correction. This function is designed to produce positive intensities. Any intensity value lower less than 0.5 is reset to be equal to 0.5. Additionally, a constant of 50 (normally) is used as a offset that it is added to the intensity values before the log transformation. The propouse of this calculation is to shrunk the log ratios to zero at the lower intensities and thus to reduce the variability of log-ratios for low intensity spots. The optimal choice for the offset is the one which makes the variability of the log-ratios as constant as possible accross the range of intensity values (Smyth, G. in BioC mailing List). If the 'half' method is chosen for Background Correction, the method will substract the chosen BACKGROUND signal to the chosen FOREGROUND signal, to produce positive corrected intensities according to the 'half' method. If the 'normexp' method is selected, then a convolution of normal and exponential distributions is fitted to foreground intensities using background intensities as a covariate, and the expected signal given the observed foreground becomes the corrected intensity. See 'limma' user guide for details.

Value

a 'RGList' object, containing in 'RGList$G' the log-2 normalized intensities

Author(s)

Pedro Lopez-Romero

References

Bolstad, B. M. (2001), Probe level quantile normalization of high density oligonucleotide array data. Unpublished Manuscript: http://bmbolstad.com/stuff/qnorm.pdf

Bolstad, B. M., Irizarry R. A., Astrand, M., and Speed, T. P. (2003), A comparison of normalization methods for high density oligonucleotide array data based on bias and variance. Bioinformatics 19, 185-193.

Smyth, G. K. (2005). Limma: linear models for microarray data. In: 'Bioinformatics and Computational Biology Solutions Using R and Bioconductor'. R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, W. Huber (eds), Springer, New York, pages 397 - 420

See Also

Agilent Feature Extraction Reference Guide http://www.Agilent.com See also 'backgroundCorrect' and 'normalizeBetweenArrays' in the limma package and 'vsn' in the vsn package.

Examples

```
## Not run:
         data(dd)
         ddNORM=BGandNorm(dd,BGmethod='half',NORMmethod='quantile',
                        foreground='MeanSignal',background='BGMedianSignal',
                        offset=50,makePLOTpre=TRUE,makePLOTpost=TRUE)
## End(Not run)
```

boxplotNegCtrl *Boxplot of Signals and Negative Controls*

Description

For each array, it shows the boxplot for the genes (red) and negative controls (green)

Usage

```
boxplotNegCtrl(RGlist, Log2, channel)
```

Arguments

RGlist	An RGlist object
Log2	logical, if TRUE it assumes that the data are in log2 scale
channel	if 'channel=R', then uses the data stored in dd$R, if channel is missing or 'channel = G' then the data stored in dd$G is used

Details

It allows the comparison of the gene signals with the signals of the negative controls.

Author(s)

Pedro Lopez-Romero

Examples

```
## Not run:
        data(dd)
        boxplotNegCtrl(dd,Log2=FALSE,channel="G")
## End(Not run)
```

BoxPlot *Boxplot*

Description

It creates a Boxplot with a matrix columns

Usage

```
BoxPlot(object, maintitle, colorfill, xlab, ylab)
```

Arguments

object	A matrix
maintitle	character to indicate the title of the graph
colorfill	color to fill the boxplot
xlab	title for the x axe
ylab	title for the y axe

Author(s)

Pedro Lopez-Romero

Examples

```
## Not run:
                data(dd)
                MMM=log2(dd$G)
                maintitle="MeanSignal"
                colorfill="green"
                BoxPlot(MMM,maintitle,colorfill,xlab="Samples",ylab="expression")
## End(Not run)
```

---

build.eset *ExpressionSet object from a RGList*

Description

It creates an 'ExpressionSet' object from a 'RGList' containing unique ProbeNames

Usage

```
build.eset(RGlist, targets, makePLOT, annotation.package)
```

Arguments

RGlist	An RGList containing normally the processed data
targets	data.frame with the targets structure
makePLOT	logical, if TRUE it makes a 'heatmap' with the 100 greater variance genes, a 'hierarchical cluster' with all the genes and a pca plot
annotation.package	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Details

It creates an ExpressionSet object from a RGList. Usually this function is applied to an RGList object containing the processed data.

Value

An ExpressionSet object

Author(s)

Pedro Lopez-Romero

See Also

write.eset

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        selSNR=which(dd$genes$ControlType==0)
        dd.aux=dd[selSNR,]
        index=which(duplicated(dd.aux$genes$ProbeName)==FALSE)
        dd.aux=dd.aux[index,]
        eset.test=build.eset(dd.aux,targets,makePLOT=TRUE,
                annotation.package="hgug4112a.db")

## End(Not run)
```

`build.mappings` *Creates data.frame with information mapped from the annotation package*

Description

Data frame with the mappings for the PROBE ID in the corresponding annotation package

Usage

```
build.mappings(esetPROC, annotation.package)
```

Arguments

`esetPROC`	Expression Set object containing processed DATA
`annotation.package`	character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Details

Creates a data.frame: by rows it contains PROBE IDs and by columns contains "ACCNUM","SYMBOL","ENTREZID", "DESCRIPTION","GO.Id" and "GO.Terms" for each probe. Mappings are extracted from the corresponding annotation package. Usually this function is applied to an Expression Set object containing the processed data

Value

data.frame

Author(s)

Pedro Lopez-Romero

See Also

`build.eset`

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        selSNR=which(dd$genes$ControlType==0)
        dd.aux=dd[selSNR,]
        index=which(duplicated(dd.aux$genes$ProbeName)==FALSE)
        dd.aux=dd.aux[index,]
        eset.test=build.eset(dd.aux,targets,makePLOT=FALSE,
                annotation.package="hgug4112a.db")

        mappings=build.mappings(eset.test,"hgug4112a.db")
        names(mappings)
## End(Not run)
```

`countFLAG` *filter.probes (internal function)*

Description

An internal function to be used by `filter.probes` For a given feature characterized by FLAG, such as IsFound, the function categorizes the number of FLAGS of one kind. For example, for a given PROBE the FLAG = 1 can be detected in 1,2,....,number Arrays. The function calculates how many probes belong to each of the different categories.

Usage

```
countFLAG(data, flag)
```

Arguments

`data`

`flag`

Author(s)

Pedro Lopez-Romero

See Also

`filter.probes`

`CV.rep.probes` *Non-control replicated Probes identication*

Description

Computes the non-control replicated probes

Usage

```
CV.rep.probes(ddDUP, annotation.package, foreground, raw.data, writeR,targets)
```

Arguments

`ddDUP`	An RGList
`annotation.package`	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'
`foreground`	a character specifying the signal from wich the CV is calculated: 'MeanSignal' or 'ProcessedSignal'
`raw.data`	logical, `TRUE` if the RGList contains the RAW data
`writeR`	logical, `TRUE` to write the REPORT FILE 'Probe.Sets.txt'
`targets`	data.frame containing the TARGET file

Details

Agilent arrays contain a number of non-control probes replicated up to ten times spread accross the array. This allows computing the of coefficient of variation) for each array. The CV is computed for every set of replicated probes. CV median is reported as the array CV. A lower CV median indicates a better array reproducibility.

Value

A txt file 'Probe.Sets.txt' that contains PROBE, number of replicates, ACCNUM code, SYMBOL code, DESCRIPTION of the gene, and probe of the arrays A boxplot that shows CV distribution computed for every set of replicated probes

Author(s)

Pedro Lopez-Romero

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        CV.rep.probes(dd,"hgug4112a.db",
                foreground="MeanSignal",raw.data=TRUE,writeR=TRUE,targets)
## End(Not run)
```

dd *data example (RGList)*

Description

Data, extracted from scanned images using Agilent Feature Extraction Software, are stored in a RGList object. This example includes 2 experimental conditions. Two replicates are considered

Usage

```
data(dd)
```

Details

A data example is provided, in which the original data have been trimmed to reduced the disk space storing. As a consequence, some of the functions regarding countint replicated probes, etc., will produce numbers that will not coincide with a 'real data'. Despite of this, the example is valid to illustrate all the function features of the functions included in the package.

Chips were scanned using the Agilent G2567AA Microarray Scanner System (Agilent Technologies) with the extended dynamic range option turned on. Image analysis and data collection were carried out using the Agilent Feature Extraction 9.1.3.1. (AFE). For the background signal calculation the AFE was set to use the spatial detrend surface value that estimate the noise due to a systematic gradient on the array and whose computation is based on a Loess algorithm. Details of how the spatial detrend algorithm works can be found in the AFE reference guide.

Data, colected with the Agilent Feature Extraction Software, are stored in a RGList object with the following components:

dd$R:	'gProcessedSignal'
dd$G:	'gMeanSignal'
dd$Rb:	'gBGMedianSignal'
dd$Gb:	'gBGUsed'
dd$targets:	'targets'
dd$genes$ProbeName:	'Probe Name'
dd$genes$GeneName:	'Gene Name'
dd$genes$SystematicName:	'Systematic Name'
dd$genes$Description:	'Description Name'
dd$genes$Sequence:	'60 bases Sequence'
dd$genes$ControlType:	'FLAG to specify the sort of feature'
dd$other$gIsWellAboveBG:	'FLAG IsWellAboveBG'
dd$other$gIsFound:	'FLAG IsFound'
dd$other$gIsSaturated:	'FLAG IsSaturated'
dd$other$gIsFeatPopnOL:	'FLAG IsFeatPopnOL'
dd$other$gIsFeatNonUnifOL:	'FLAG IsFeatNonUnifOL'
dd$other$chr_coord:	'CHR coordinate (obtained from Agilent data files)'

Later, the if an annotation package exists, the fields 'SystematicName', 'GeneName' and 'Description' are replaced, respectively, by the corresponding ACCNUM, SYMBOL and DESCRIPTION obtained from the annotation package.

See Also

read.targets targets

Examples

```
## Not run:
        data(dd)
        head(data)
## End(Not run)
```

---

ensembl.htmlpage *genes.rpt.agi (Internal function)*

Description

Internal function to be used by genes.rpt.agi

Usage

```
ensembl.htmlpage(probe.ids, probe.chr, filename, annotation.package, title, othe
```

Arguments

probe.ids

probe.chr

filename

annotation.package
: a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

title

othernames

table.head

table.center

Details

It writes an html file with a link to the ENSEMBL data base for each probe in the input.

Value

An html file

Author(s)

Pedro Lopez-Romero

See Also

genes.rpt.agi

Examples

```
## Not run:
        data(dd)
        library(hgug4112a.db)
        PROBE_ID=dd$genes$ProbeName[200:210]
        probe.chr=dd$other$chr_coord[200:210,1]
        GENE_ID = unlist(lookUp(PROBE_ID, "hgug4112a.db", "ACCNUM") )
        gene.sym=lookUp(PROBE_ID,"hgug4112a.db","SYMBOL")
        filename=paste("Gen.Sets","example","html",sep=".")
        title="Replicated Gene"
        head <- c("PROBE ID","ACCNUM","SYMBOL","probe chr coord")

        ensembl.htmlpage(PROBE_ID,probe.chr,filename,"hgug4112a.db",
                                title, table.head=head,table.center = TRUE,
                                other=list(unlist(GENE_ID),unlist(gene.sym),unlist(probe.

## End(Not run)
```

---

filter.control *filter.probes (Internal function)*

Description

An internal function to be used by `filter.probes` An internal function to be used by `CV.rep.probes`

Usage

```
filter.control(ddNORM, ManuelaGO, targets, annotation.package)
```

Arguments

`ddNORM`	an RGlist object
`ManuelaGO`	logical, if a known annotation package is available then it is `TRUE`
`targets`	data.frame with the target structure
`annotation.package`	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Details

It eliminates the array internal controls

Value

An RGlist with the internal controls filtered out It also writes an output file (RawDataNOCtrl.txt) that contains the data set with the controls filtered out and an output file (RawDataNOCtrlWABKGandISF.txt) that contains the data with the flags 'wellAboveBG' and 'IsFound'. This flags both take the value =1 if the signal is well above BG and Is found respectively. See 'filter.wellaboveBG' and 'filter.probes'

Author(s)

Pedro Lopez-Romero

See Also

filter.probes

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        ddFILT=filter.control(dd,TRUE,targets,"hgug4112a.db")
        dim(ddFILT)

## End(Not run)
```

---

filterFLAGall *filter.probes (Internal function)*

Description

An internal function to be used by filter.PopnOL An internal function to be used by filter.NonUnifOL

Usage

```
filterFLAGall(flag, GErep, nGE, minFLAG, limFLAGall)
```

Arguments

flag

GErep

nGE

minFLAG

limFLAGall

---

filterFLAG *probes.filter (Internal function)*

Description

An internal function to be used by filter.wellaboveBG An internal function to be used by filter.isfound An internal function to be used by filter.saturated

Usage

```
filterFLAG(flag, GErep, nGE, minFLAG, limSNR)
```

Arguments

flag

GErep

nGE

minFLAG

limSNR

Author(s)

Pedro Lopez-Romero

---

filter.isfound *probes.filter (Internal function)*

Description

An internal function to be used by filter.probes

Usage

```
filter.isfound(ddFILT, limISF, ManuelaGO, targets, annotation.package)
```

Arguments

ddFILT	
limISF	for a given spot xi accros samples, is the minimum in a experimental condition with a isfound-FLAG = 1 (Is Found)
ManuelaGO	logical, if a known annotation package is available then it is TRUE
targets	data.frame with the target structure
annotation.package	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Details

It eliminates signals that were not found. The filtering is based on the FLAG given by the AFE as follows: For a feature = xi accross all the samples, we demand that at least p spots for that feature, in at least one experimental condition, had a quantification flag denoting that the signal is found. A spot is considered Found if two conditions are true: 1) the difference between spot signal and the local background signal is more than 1.5 times the local background noise and 2) the spot diameter is at least 0.30 times the nominal spot diameter. A Boolean variable is used to flag found features. 1 = IsFound

Value

An RGlist with probes that are found (according to the correspondind AFE flag and the filtering options). It also writes an output file (IsNOTFound.txt) that contains probes that were filtered out because were NOT FOUND.

Author(s)

Pedro Lopez-Romero

See Also

filter.probes

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        ddFILT=filter.isfound(dd,75,TRUE,targets,"hgug4112a.db")
        dim(ddFILT)
## End(Not run)
```

---

filterNAS *probes.filter (Internal function)*

Description

An internal function to be used by filter.nas

Usage

```
filterNAS(signal, GErep, nGE, limNAS)
```

Arguments

signal

GErep

nGE

limNAS

Author(s)

Pedro Lopez-Romero

filter.nas *probes.filter (Internal function)*

Description

An internal function to be used by `filter.probes`

Usage

```
filter.nas(ddFILT, limNAS, targets)
```

Arguments

`ddFILT`

`limNAS`

`targets`

Author(s)

Pedro Lopez-Romero

filter.NonUnifOL *probes.filter (Internal function)*

Description

An internal function to be used by `filter.probes`

Usage

```
filter.NonUnifOL(ddFILT, limNonUnifOL, ManuelaGO, targets, annotation.package)
```

Arguments

`ddFILT`	an RGlist object
`limNonUnifOL`	for a given feature xi accros samples, is the maximum in a experimental condition with a saturation-FLAG = 1 (Is Non Uni OL)
`ManuelaGO`	logical, if a known annotation package is available then it is `TRUE`
`targets`	data.frame with the target structure
`annotation.package`	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Details

To keep good quality probes, we filtered out probes that has more than y in at least one experimental condition with a flag indicating the presence of non-uniformity outlier. A feature is non-uniform outlier if its pixel noise exceeds a threshold established for a uniform feature. 1 indicates Feature is a non-uniformity outlier.

Value

An RGlist with probes that are not non-uniformity Outlier (according to the correspondind AFE flag and the filtering options). It also writes an output file (IsFeatNonUnifOL.txt) that contains probes that were filtered out because they were considered non-uniformity Outlier.

Author(s)

Pedro Lopez-Romero

See Also

`filter.probes`

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        ddFILT=filter.NonUnifOL(dd,25,TRUE,targets,"hgug4112a.db")
        dim(ddFILT)
## End(Not run)
```

---

filter.PopnOL *probes.filter (Internal function)*

Description

An internal function to be used by `filter.probes`

Usage

```
filter.PopnOL(ddFILT, limPopnOL, ManuelaGO, targets, annotation.package)
```

Arguments

`ddFILT`	an RGlist object
`limPopnOL`	for a given feature xi accros samples, is the maximum in a experimental condition with a saturation-FLAG = 1 (Is Pop OL)
`ManuelaGO`	logical, if a known annotation package is available then it is `TRUE`
`targets`	data.frame with the target structure
`annotation.package`	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Details

To keep good quality probes, we filtered out probes that had more than y in at least one experimental condition with a flag indicating presence of Population Outlier. A feature is a population outlier if its signal is lower than a lower threshold or higher than an upper threshold determined using a multiplier (1.42) times the interquartile range of the population. 1 indicates Feature is a population outlier.

Value

An RGlist with probes that are not Population Outlier (according to the correspondind AFE flag and the filtering options). It also writes an output file (IsFeatPopnOL.txt) that contains probes that were filtered out because they were considered Population Outlier.

Author(s)

Pedro Lopez-Romero

See Also

filter.probes

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        ddFILT=filter.PopnOL(dd,25,TRUE,targets,"hgug4112a.db")
        dim(ddFILT)
## End(Not run)
```

---

filter.probes *Filtering out probes*

Description

Filter out probes according to their Quality Flag

Usage

```
filter.probes(ddNORM,control = NULL, wellaboveBG = NULL, isfound = NULL, wellabo
```

Arguments

ddNORM	An RGList in log2 scale to be FILTERED out according to a Quality FLAG
control	LOGICAL: If True it removes controls
wellaboveBG	LOGICAL: If True it filters by Well Above BG FLAG
isfound	LOGICAL: If True it filters by Is Found FLAG
wellaboveNEG	LOGICAL: If True it filters by Well Above NEG CTRLS
sat	LOGICAL: If True it filters by Is Saturated FLAG
PopnOL	LOGICAL: If True it filters by Population Outlier FLAG
NonUnifOL	LOGICAL: If True it filters by Non Uniform Outlier FLAG
nas	LOGICAL: If True it removes NAs
limWellAbove	for a given spot xi accros samples, is the minimum in a experimental condition with a wellaboveBG-FLAG = 1 (Is Well Above BG)
limISF	for a given feature xi accros samples, is the minimum in a experimental condition with a isfound-FLAG = 1 (Is Found)

limNEG	for a given feature xi accros samples, is the minimum in a experimental condition with a intensity > Limit established for negative controls (Mean + 1.5 x SD)
limSAT	for a given feature xi accros samples, is the minimum in a experimental condition with a saturation-FLAG = 0 (Non Saturated)
limPopnOL	for a given feature xi accros samples, is the minimum in a experimental condition with a saturation-FLAG = 1 (Is Pop OL)
limNonUnifOL	for a given feature xi accros samples, is the minimum in a experimental condition with a saturation-FLAG = 1 (Is Non Uni OL)
limNAS	for a given feature xi accros samples, is the minimum in a experimental condition
makePLOT	LOGICAL: If True it makes QC graphs filtering
annotation.package	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'
flag.counts	LOGICAL: If True it runs the countFLAG function
targets	data.frame with the target structure

Details

Agilent Feature Extraction software provides a flag for each spot that identifies different quantification errors of the signal. Quantification flags were used to filter out signals that did not reach a minimum established criterion of quality. Data were filtered at a probe level according to the following criteria. a- To keep features within the dynamic range of the scanner: For a spot = xi accross all the samples, we demand that at least p probes of the spot xi in at least one experimental condition had a quantification flag denoting that the signal is distinguishable from background. The same criterion is applied independently for the 'IsFound' flag and for signal saturation. b- To keep good quality features, we filtered out probes that had more than y in at least one experimental condition flagged as Outliers.

Value

The function returns a RGList containing with the FILTERED data eliminated In order to allow the tracking of features that may have been filtered out from the original raw data, the following files are given:

RawDataNOCtrl.txt: contains all the features included in the array once the internal controls have been removed. Internal controls are removed prior to any preprocessing step.

IsNOTWellAboveBG.txt: contains the features that have been filtered out because they are not distinguishable from the local background signal.

We uses a Boolean flag indicating if a feature is WellAbove Background (Flag = 1) or not (Flag = 0). A feature reaches a Flag = 1 if IsPosAndSignif and additionally the gBGSubSignal is greater than 2.6*g(r)BG_SD.

IsPosAndSignif uses a Boolean flag, established via a 2-sided t-test, indicates whether the mean signal of a feature is greater than the corresponding background. 1 indicates feature is positive and significant above background

IsNOTFound.txt: contains features that have been filtered out because were NOT FOUND. A feature is considered Found if two conditions are true: 1- the difference between the feature signal and the local background signal is more than 1.5 times the local background noise and 2- the spot diameter is at least 0.30 times the nominal spot diameter. A Boolean variable is used to flag found features. 1 = IsFound

IsSaturated.txt: contains the features that are saturated. A feature is saturated IF 50 threshold. 1 = Saturated

IsFeatNonUnifOL.txt: contains the features that are considered Non Uniformity Outlier. A feature is non-uniform if the pixel noise of feature exceeds a threshold established for a uniform feature. 1 indicates Feature is a non-uniformity outlier.

IsFeatPopnOL.txt: contains the features that are considered Population Outlier. A feature is a population outlier if its signal intensity is lower than a lower threshold or exceeds an upper threshold determined using a multiplier (1.42) times the interquartile range of the population. 1 indicates Feature is a population outlier

IsNOTWellAboveNEG.txt: Besides, for each feature we can demand a minimum signal value that have to be reached at least for a p in one of the experimental conditions. The minimum limit has been established as Mean Negative Controls + 1.5*(Std. dev.Negative Controls). Normally, after filtering by WellAboveBG and IsFound criteria, all probes are well above negative controls.

In addition to all files indicated above we have added ACCNUM, GENE SYMBOL, ENTREZID reference and gene DESCRIPTION corresponding to each manufacturer probe code in the corresponding annotation package.

Author(s)

Pedro Lopez-Romero

Examples

```
## Not run:
         data(dd)
         data(targets)
         library(hgug4112a.db)
         ddNORM=BGandNorm(dd,BGmethod='half',NORMmethod='quantile',
                        foreground='MeanSignal',background='BGMedianSignal',
                        offset=50,makePLOTpre=FALSE,makePLOTpost=FALSE)
         ddFILT=filter.probes(ddNORM,
                control=TRUE,
                wellaboveBG=TRUE,
                isfound=TRUE,
                wellaboveNEG=TRUE,
                sat=TRUE,
                PopnOL=TRUE,
                NonUnifOL=TRUE,
                nas=TRUE,
                limWellAbove=75,
                limISF=75,
                limNEG=75,
                limSAT=75,
                limPopnOL=75,
                limNonUnifOL=75,
                limNAS=100,
                makePLOT=TRUE,annotation.package="hgug4112a.db",flag.counts=TRUE,targets)
## End(Not run)
```

`filter.saturated` *probes.filter (Internal function)*

Description

An internal function to be used by `filter.probes`

Usage

```
filter.saturated(ddFILT, limSAT, ManuelaGO, targets, annotation.package)
```

Arguments

`ddFILT`	an RGlist object
`limSAT`	for a given feature xi accros samples, is the minimum in a experimental condition with a saturation-FLAG = 0 (Non Saturated)
`ManuelaGO`	logical, if a known annotation package is available then it is `TRUE`
`targets`	data.frame with the target structure
`annotation.package`	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Details

It eliminates saturated signals. The filtering is based on the FLAG given by the AFE as follows: For a feature = xi accross all the samples, we demand that at least p spots for that feature,in at least one experimental condition, had a quantification flag denoting that the signal is not saturated. AFE produces a Boolean flag indicating whether a spot is saturated (Flag = 1) or not saturated (Flag = 0). A spot is saturated IF 50 threshold.

Value

An RGlist with probes that are not saturated (according to the correspondind AFE flag and the filtering options). It also writes an output file (IsSaturated.txt) that contains probes that were saturated.

Author(s)

Pedro Lopez-Romero

See Also

`filter.probes`

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        ddFILT=filter.saturated(dd,75,TRUE,targets,"hgug4112a.db")
        dim(ddFILT)

## End(Not run)
```

`filter.wellaboveBG` *probes.filter (Internal function)*

Description

An internal function to be used by `filter.probes`

Usage

```
filter.wellaboveBG(ddFILT, limWellAbove, ManuelaGO, targets, annotation.package)
```

Arguments

`ddFILT`	an RGlist object
`limWellAbove`	for a given feature xi accros samples, is the minimum in an experimental condition with a wellaboveBG-FLAG = 1 (Is Well Above BG)
`ManuelaGO`	logical, if a known annotation package is available then it is `TRUE`
`targets`	data.frame with the target structure
`annotation.package`	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Details

It eliminates signals that were not distinguishable from the local background signal. The filtering is based on the FLAG given by the AFE as follows: For a feature = xi accross all the samples, we demand that at least p spots for that feature, in at least one experimental condition, had a quantification flag denoting that the signal is distinguishable from background. AFE produces a Boolean flag indicating if a spot is WellAbove Background (Flag = 1) or not (Flag = 0). A spot reaches a Flag = 1 if IsPosAndSignif and additionally the gBGSubSignal is greater than 2.6*g(r)BG_SD. IsPosAndSignif uses a Boolean flag, established via a 2-sided t-test, indicates if the mean signal of a spot is greater than the corresponding background. 1 indicates spot is positive and significant above background.

Value

An RGlist with probes that are above BG (according to the correspondind AFE flag and the filtering options). It also writes an output file (IsNOTWellAboveBG.txt) that contains the probes that were filtered out because they were not distinguishable from the local background signal.

Author(s)

Pedro Lopez-Romero

See Also

`filter.probes`

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        ddFILT=filter.wellaboveBG(dd,75,TRUE,targets,"hgug4112a.db")
        dim(ddFILT)
## End(Not run)
```

---

filter.wellaboveNEG

probes.filter (Internal function)

Description

An internal function to be used by `filter.probes`

Usage

```
filter.wellaboveNEG(ddFILT, ddNORM, limNEG, SDtimes, ManuelaGO, targets, annotat
```

Arguments

`ddFILT`	A RGlist in log2 scale, normally after Normalization and other filtering steps
`ddNORM`	An RGlist normally containing NORMALIZED data in log2 scale
`limNEG`	for a given feature xi accros samples, is the minimum in a experimental condition with a intensity > Limit established for negative controls (Mean + 1.5 x SD)
`SDtimes`	1.5 in 'Mean + 1.5 x SD'. It is fixed to 1.5 in 'filter.probes'
`ManuelaGO`	logical, if a known annotation package is available then it is `TRUE`
`targets`	data.frame with the target structure
`annotation.package`	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Details

For each feature we can demand a minimum signal value that have to be reached at least for a p in one of the experimental conditions. The minimum limit is established as Mean Negative Controls + 1.5*(Std. dev.Negative Controls). Normally, after filtering by the WellAboveBG and IsFound criteria, all probes are well above negative controls.

Value

An RGlist with signals that are above NEG controls (according to the correspondind AFE flag and the filtering options). It also writes an output file (IsNOTWellAboveNEG.txt) that contains probes that were filtered out because they were not distinguishable from negative controls.

Author(s)

Pedro Lopez-Romero

See Also

filter.probes

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        ddNORM=BGandNorm(dd,BGmethod='half',NORMmethod='quantile',
                         foreground='MeanSignal',background='BGMedianSignal',
                         offset=50,makePLOTpre=FALSE,makePLOTpost=FALSE)
        ddFILT=filter.wellaboveBG(ddNORM,75,TRUE,targets,"hgug4112a.db")
        ddFILT2=filter.wellaboveNEG(ddFILT,ddNORM,75,SDtimes=1.5,TRUE,targets,"hgug4112a.
        dim(ddFILT2)
## End(Not run)
```

---

filterWellAboveSIGNALv2

probes.filter (Internal function)

Description

An internal function to be used by filter.wellaboveNEG

Usage

```
filterWellAboveSIGNALv2(flag, GErep, nGE, Limit, limNEG)
```

Arguments

flag

GErep

nGE

Limit

limNEG

Author(s)

Pedro Lopez-Romero

`genes.rpt.agi` *Genes interrogated for different probes*

Description

Some Genes are interrogated by different probes at different positions. This function identifies different probes interrogating the same gene. This group of probes is called a GENE SET.

Usage

```
genes.rpt.agi(dd, annotation.package, raw.data, WRITE.html, REPORT)
```

Arguments

`dd`	An RGList: raw data or Processed data
`annotation.package`	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'
`raw.data`	logical, if `TRUE` it uses RAW data, otherwise it uses PROCESSED data
`WRITE.html`	logical, if `TRUE` it writes an html output
`REPORT`	logical, if `TRUE`it generates an information REPORT

Value

Generates an 'html' file and a 'txt' file with information about genes that are interrogated for different probes at different chr positions. The 'html' file includes a link to the ENSEMBL data base, exactly to the chr region where the 60 base agilent probe claims to be located. Chr coordinates are taken directly from the agilent files (data files obtained after image scanning and using Agilent Feature Extraction)

Author(s)

Pedro Lopez-Romero

Examples

```
data(dd)
library(hgug4112a.db)
genes.rpt.agi(dd,"hgug4112a.db",raw.data=TRUE,
        WRITE.html=TRUE,REPORT=TRUE)
```

getTDRows3 *ensembl.htmlpage: (Internal function)*

Description

An internal function to be used by ensembl.htmlpage

Usage

```
getTDRows3(probe.ids, probe.chr, annotation.package)
```

Arguments

probe.ids

probe.chr

annotation.package
: a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

Author(s)

Pedro Lopez-Romero

gsea.files *Creates files for GSEA*

Description

Creates data and phenotype files for GSEA http://www.broad.mit.edu/gsea

Usage

```
gsea.files(eset, targets, annotation.package)
```

Arguments

eset
: Expression Set object containing processed DATA

annotation.package
: character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

targets
: data.frame with the targets structure

Author(s)

Pedro Lopez-Romero

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        selSNR=which(dd$genes$ControlType==0)
        dd.aux=dd[selSNR,]
        index=which(duplicated(dd.aux$genes$ProbeName)==FALSE)
        dd.aux=dd.aux[index,]
        eset.test=build.eset(dd.aux,targets,makePLOT=FALSE,
                annotation.package="hgug4112a.db")

        gsea.files(eset.test,targets,annotation.package="hgug4112a.db")
## End(Not run)
```

HeatMap *HeatMap*

Description

Creates a HeatMap graph using the 'heatmap.2' function

Usage

```
HeatMap(object, size, maintitle)
```

Arguments

object	A expression Matrix
size	number of highest variance genes to be considered in the plot
maintitle	title of the plot

Author(s)

Pedro Lopez-Romero

See Also

heatmap.2

Examples

```
## Not run:
        data(dd)
        selSNR=which(dd$genes$ControlType==0)
        dd.aux=dd[selSNR,]
        index=which(duplicated(dd.aux$genes$ProbeName)==FALSE)
        dd.aux=dd.aux[index,]
        maintitle="100 High Var"
        HeatMap(dd.aux$G,100,maintitle)
## End(Not run)
```

`hierclus` *Hierarchical clustering*

Description

Hierarchical cluster of samples using the 'hclust' function

Usage

```
hierclus(object, GErep, methdis, methclu,sel, size)
```

Arguments

`object`	An expression Matrix
`GErep`	Numerical vector that relates each sample with its experimental condition
`methdis`	the distance measure to be used. Options are 'euclidean' and 'pearson'. see 'dist' function
`methclu`	the agglomeration method to be used by the 'hclust' function
`sel`	logical, if `TRUE` selects the 'size' highest variance genes for the plot
`size`	selects the 'size' highest variance genes for the plot if 'sel=TRUE'

Author(s)

Pedro Lopez-Romero

See Also

hclust,dist

Examples

```
## Not run:
        data(dd)
        data(targets)
        GErep=targets$GErep
        selSNR=which(dd$genes$ControlType==0)
        dd.aux=dd[selSNR,]
        index=which(duplicated(dd.aux$genes$ProbeName)==FALSE)
        dd.aux=dd.aux[index,]
        hierclus(dd.aux$G,GErep,methdis="euclidean",
        methclu="complete",sel=FALSE,100)
## End(Not run)
```

MVAplotMEDctrl *MVA plot*

Description

For each array, the M value is computed for every spot as the difference between the spot intensity in the array and the averaged intensity for that feature over the whole set of arrays. Every kind of feature is identified with different color (gene, controls, etc ...)

Usage

```
MVAplotMEDctrl(RGlist, maintitle, channel = NULL)
```

Arguments

RGlist	An RGlist object
maintitle	character to indicate the title of the graph
channel	if 'channel=R', then uses the data stored in dd$R, if channel is missing or 'channel = G' then the data stored in dd$G is used

Author(s)

Pedro Lopez-Romero

Examples

```
data(dd)
par(mfrow=c(1,1),ask=TRUE)
MVAplotMEDctrl(dd,"MVA example",channel="G")
```

MVAplotMED *MVA plot*

Description

For each array, the M value is computed for every spot as the difference between the spot intensity in the array and the averaged intensity for that feature over the whole set of arrays

Usage

```
MVAplotMED(object, colorfill, maintitle)
```

Arguments

object	An expression matrix transformed to log2 scale
colorfill	color of the plot
maintitle	title of the plot

Author(s)

Pedro Lopez-Romero

Examples

```
## Not run:
        data(dd)
        maintitle="MVA example"
        par(mfrow=c(1,1),ask=TRUE)
        MVAplotMED(log2(dd$G),colorfill="red",maintitle)
## End(Not run)
```

---

paste.character *paste characters (Internal function)*

Description

An internal function to be used by genes.rpt.agi An internal function to be used by build.mappings

Usage

```
paste.character(paux)
```

Arguments

paux

Author(s)

Pedro Lopez-Romero

PCAplot *PCA plot*

Description

It is a wrapper for the 'plotPCA' of the 'affycoretools' package

Usage

```
PCAplot(eset, targets)
```

Arguments

eset	An Expression Set object
targets	data.frame with the target structure

Author(s)

Pedro Lopez-Romero

Examples

```
## Not run:
        data(dd)
        data(targets)
        selSNR=which(dd$genes$ControlType==0)
        dd.aux=dd[selSNR,]
        index=which(duplicated(dd.aux$genes$ProbeName)==FALSE)
        dd.aux=dd.aux[index,]
        eset.test=build.eset(dd.aux,targets,makePLOT=FALSE,
                annotation.package="hgug4112a.db")
        PCAplot(eset.test,targets)
## End(Not run)
```

---

plotDensity *Density Plots of Intensity Signals*

Description

Creates a density plot with arrays intensities

Usage

```
plotDensity(object, maintitle)
```

Arguments

object	An expression matrix, in log2 scale
maintitle	title of the plot

Author(s)

Pedro Lopez-Romero

Examples

```
## Not run:
        data(dd)
        maintitle="Density Plot example"
        plotDensity(log2(dd$G),maintitle)
## End(Not run)
```

`read.AgilentFE` *Read Agilent Feature Extraction *.txt data files*

Description

Read the data files generated by the Agilent Feature Extraction image analysis software

Usage

```
read.AgilentFE(targets, makePLOT)
```

Arguments

`targets`	A data frame that specifies experimental conditions under which each sample has been obtained.
`makePLOT`	logical, if `TRUE` the function display boxplots and density plots of foreground 'ProcessedSignal' and 'MeanSignal' and background 'BGMedianSignal' and 'BGUsed'

Details

The function reads the *.txt files generated by the AFE Software using the 'read.maimages' function of 'limma' package.

Data, colected with the Agilent Feature Extraction Software, are stored in a RGList object with the following components:

dd$R:	'gProcessedSignal'
dd$G:	'gMeanSignal'
dd$Rb:	'gBGMedianSignal'
dd$Gb:	'gBGUsed'
dd$targets:	'targets'
dd$genes$ProbeName:	'Probe Name'
dd$genes$GeneName:	'Gene Name'
dd$genes$SystematicName:	'Systematic Name'
dd$genes$Description:	'Description Name'
dd$genes$Sequence:	'60 bases Sequence'
dd$genes$ControlType:	'FLAG to specify the sort of feature'
dd$other$gIsWellAboveBG:	'FLAG IsWellAboveBG'
dd$other$gIsFound:	'FLAG IsFound'
dd$other$gIsSaturated:	'FLAG IsSaturated'
dd$other$gIsFeatPopnOL:	'FLAG IsFeatPopnOL'
dd$other$gIsFeatNonUnifOL:	'FLAG IsFeatNonUnifOL'
dd$other$chr_coord:	'CHR coordinate (from Agilent data files)

'MeanSignal' is the spot Raw mean signal. 'ProcessedSignal' is the signal processed by the AFE software analysis software (AFE). It contains the Multiplicatively Detrend Bacground Substracted Signal if detrending option is selected and it helps. If the detrending does not help, the 'ProcessedSignal' will be the Bacground Subtracted Signal. 'BGMedianSignal' is the Median local background signal. 'BGUsed' depends on the AFE settings for the type of background method and the setting for the spatial detrend. Usually, the Background Signal Used is the sum of the local bac-

ground + the spatial detrending surface value computed by the AFE software. To view the values used to calculate this variable using different bakground signals and settings of spatial detrend and global background adjust, see Table 33 on page 213 in the AFE reference guide.

Later, if a BioC annotation package is provided, 'SystematicName', 'GeneName' and 'Description' fields are replaced, respectively, by the corresponding ACCNUM, SYMBOL and DESCRIPTION obtained from the annotation package.

Value

An RGList containing:

`RGList$Rf` matrix, 'gProcessedSignal'

`RGList$Gf` matrix, 'gMeanSignal'

`RGList$Rb` matrix, 'gBGMedianSignal'

`RGList$Gb` matrix, 'gBGUsed'

`RGList$targets`
data.frame, 'FileName'

`RGList$genes$ProbeName`
character, 'AGilent Probe Name'

`RGList$genes$GeneName`
character, 'Symbol Gene Name'

`RGList$genes$SystematicName`
character, 'Systematic Gene Name'

`RGList$genes$Description`
character, 'Gene Description'

`RGList$genes$Sequence`
character, 'Sequence of the 60 mer probe'

`RGList$genes$ControlType`
integer, '0'= Feature, '1'= Positive control, '-1'= Negative control

`RGList$other$gIsWellAboveBG`
matrix, FLAG to classify signal if 'IsWellAboveBG=1' or 'not=0'

`RGList$other$gIsFound`
matrix, FLAG to classify signal if 'IsFound=1' or 'not=0'

`RGList$other$gIsSaturated`
matrix, FLAG to classify signal if 'IsSaturated = 1' or 'not=0'

`RGList$other$gIsFeatPopnOL`
matrix, FLAG to classify signal if 'IsFeatPopnOL = 0' or 'not=1'

`RGList$other$gIsFeatNonUnifOL`
matrix, FLAG to classify signal if 'gIsFeatNonUnifOL = 0' or 'not=1'

`RGList$other$chr_coord`
matrix, chr coordinates for the 60 mer Agilent probe

Author(s)

Pedro Lopez-Romero

References

Agilent Feature Extraction Reference Guide http://www.Agilent.com

See Also

read.targets dd targets

Examples

```
## Not run:
        dd=read.AgilentFE(targets,makePLOT=TRUE)
## End(Not run)
        data(dd)
```

---

read.targets *read the target file*

Description

The target file is a txt file where every input file (array, sample) is attached to a experimental condition

Usage

```
read.targets(infile)
```

Arguments

infile	name of the target file, for instance 'targets.txt'

Details

Intention of data preprocessing using Agi4x44PreProces is to carry out a subsequent statistical analysis searching for genes that are differentially expressed between different experimental conditions. First, in the 'target file' we specify the experimental conditions under which the data have been generated. From this primary statistical analysis, one can be perform other analysis such as Functional Enrichment analysis, etc ... The target file MUST contain the following columns: -FileName : Name of the array file, xxx.txt -Treatment : Treatment of the file xxx.txt -GErep : Treatment of the file xxx.txt in numeric code, from 'treatment-1' to 'treatment-N' Other OPTIONAL columns can be -Subject : If treatment are paired by 'Subject': -Array : If we want to take into account that some samples have been hibridyzed on the same Agi4x44 platform

Value

A 'data.frame' containing by columns the columns specified in targets.txt

Author(s)

Pedro Lopez-Romero

See Also

read.table

Examples

```
## Not run:
        targets=read.targets(infile="targets.txt")
## End(Not run)
        data(targets)
```

RLE *Relative Log Expression*

Description

RLE: Relative Log Expression

Usage

```
RLE(object, maintitle, colorfill)
```

Arguments

object	An expression matrix, in log2 scale
maintitle	title of the plot
colorfill	color of the plot

Details

Each Boxplot corresponds to a sample and displays the Relative Log Expression computed for every spot in the array as the difference between the spot intensity and the median intensity for the same feature accros all the arrays. Since majority of the spots are expected not to be differentially expressed, the plot shoud show boxplots centered around zero and all of them having the aproximately the same dispersion. An array showing greater dispersion than the other, or being not centered at zero could have quality problems.

Author(s)

Pedro Lopez-Romero

References

Boldstad B.M., Collin F., Brettschneider J., Simpson, K., Cope L., Irizarry R. A., Speed T. P. Quality Assesement of Affymetrix GeneChip Data. In Bioinformatics and Computational Biology Solutions Using R and Bioconductor. (eds.) Gentleman R., Carey V. J., Huber W., Irizarry R. A., Dudoit S. (2005). Springer.

Examples

```
## Not run:
                data(dd)
                maintitle="RLE example "
                colorfill="orange"
                RLE(log2(dd$G),maintitle,colorfill)
## End(Not run)
```

subCHR *chr coordinates edition (Internal function)*

Description

Given a chr coordinate for a probe, it creates a character vector containing the number of the chr, the start base and the end base

Usage

```
subCHR(pb.chr)
```

Arguments

```
pb.chr
```

Value

Returns a vector 'out' with the following components

`out[1]`	chr number
`out[2]`	start coordinate
`out[3]`	end coordinate

Author(s)

Pedro Lopez-Romero

summarize.probe *Replicated Probes Summarization*

Description

Computes Median of Replicated non-control probes

Usage

```
summarize.probe(ddFILT, makePLOT, targets)
```

Arguments

`ddFILT`	RGlist, usually containing Normalized and Filtered data
`makePLOT`	LOGICAL: If True it makes graphs
`targets`	data.frame with the target structure

Details

Normally, the Agilent 4 x 44 chips contain a set of non-control probes that are replicated up to ten times. These probes are spread over the chip and allow measuring the chip reproducibility in terms of the coefficient of variation (of the array. A lower CV median indicates a better reproducibility of the array. It uses an RGList as an input and it produces another RGList where each set of replicated non-control probes have been collapsed into a single value computed as the median of the probes intensities belonging to the same set. Normally, the input RGList is the 'filtered data', but other RGList can be used as inputs. This is an optional step.

Value

RGList

Author(s)

Pedro Lopez-Romero

See Also

`CV.rep.probes`

Examples

```
## Not run:
        data(targets)
        data(dd)
        ddPROC=summarize.probe(dd,makePLOT=TRUE,targets)
## End(Not run)
```

`targets` *Example of target file*

Description

Example of target file

Usage

```
data(targets)
```

Format

A data frame with 4 observations on the following 5 variables.

`FileName` names of the Files `Ast.txt` `Bst.txt` `Aunst.txt` `Bunst.txt`

`Treatment` Assigns level for Treatment Effect to each File (mandatory)

`GErep` a numeric vector tha numerates the FACTOR of the Treatment Effect (mandatory)

`Subject` Assigns level for Subject Effect to each file (optional)

`Array` Assigns level for Array Effect to each file (optional)

Details

It is a tab-delimited text format file. The target file is created with the intention of carrying out a differential expression analysis in future steps using 'limma'. Therefore, here is where factors that are going to be included in the linear model that is fitted to each gen are specified. Targets file assigns each image data file to one of the experimental conditions of the experiment. First column 'FileName' is mandatory and includes image data files names. Second column 'Treatment' is also mandatory and includes treament effect. Third column 'GErep' is also mandatory, and includes the Treatment effect in a numeric code, from 1 to n, being n the number of Treatment effect levels. The rest of the columns are optional, and they can include other factors that we want to specify in the model, such as 'Subject', 'Array', and so on.

References

Gordon K. Smyth, M. Ritchie, N. Thorne, J. Wettenhall (2007). limma: Linear Models for Microarray Data User's Guide.

Examples

```
## Not run:
        data(targets)
## End(Not run)
```

`write.control.out` *probes.filter (Internal function)*

Description

internal function to be used by `filter.control`

Usage

```
write.control.out(ddFILT, selSNR, annotation.package, ManuelaGO, targets)
```

Arguments

`ddFILT`

`selSNR`

`annotation.package`
: a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

`ManuelaGO`

`targets`

Author(s)

Pedro Lopez-Romero

`write.eset` *Writes the expression data matrix of an ExpressionSet object in a file*

Description

Writes the expression data matrix of an ExpressionSet object in a file.

Usage

```
write.eset(eset, ddPROC, annotation.package, targets)
```

Arguments

`eset`	An Expression object, normally containing the procesed data
`ddPROC`	An RGList object, normally containing the processed data
`annotation.package`	a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'
`targets`	data.frame with the targets structure

Details

Writes the expression data matrix of an ExpressionSet object in a file. It also writes out the mappings of the Agilent PROBE ID with the ACCNUM, SYMBOL, ENTREZID and DESCRIPTION fields, using the corresponding annotation package

Author(s)

Pedro Lopez-Romero

See Also

`build.eset`

Examples

```
## Not run:
        data(dd)
        data(targets)
        library(hgug4112a.db)
        selSNR=which(dd$genes$ControlType==0)
        dd.aux=dd[selSNR,]
        index=which(duplicated(dd.aux$genes$ProbeName)==FALSE)
        dd.aux=dd.aux[index,]
        eset.test=build.eset(dd.aux,targets,makePLOT=TRUE,
                annotation.package="hgug4112a.db")

        write.eset(eset.test,dd.aux,"hgug4112a.db",targets)

## End(Not run)
```

`write.filt.out` *probes.filter (Internal function)*

Description

internal function to be used by `filter.wellaboveBG` internal function to be used by `filter.isfound` internal function to be used by `filter.saturated` internal function to be used by `filter.PopnOL` internal function to be used by `filter.NonUnifOL` internal function to be used by `filter.wellaboveNEG`

Usage

```
write.filt.out(ddFILT, selSNR, ManuelaGO, annotation.package, outfile, FLAG, tar
```

Arguments

`ddFILT`

`selSNR`

`ManuelaGO`

`annotation.package`
: a character specifying the AGI annotation package: 'hgug4112a.db','mgug4122a.db'

`outfile`

`FLAG`

`targets`

Author(s)

Pedro Lopez-Romero

Index