

Package ‘iCheck’

September 23, 2026

Type Package

Title QC Pipeline and Data Analysis Tools for High-Dimensional
Illumina mRNA Expression Data

Version 1.42.0

Date 2016-11-17

Author Weiliang Qiu [aut, cre],
Brandon Guo [aut, ctb],
Christopher Anderson [aut, ctb],
Barbara Klanderman [aut, ctb],
Vincent Carey [aut, ctb],
Benjamin Raby [aut, ctb]

Maintainer Weiliang Qiu <stwxq@channing.harvard.edu>

Depends R (>= 3.2.0), Biobase, lumi, gplots

Imports stats, graphics, preprocessCore, grDevices, randomForest,
affy, limma, parallel, GeneSelectMMD, rgl, MASS, lmtest,
scatterplot3d, utils

biocViews GeneExpression, DifferentialExpression, Microarray,
Preprocessing, DNAMethylation, OneChannel, TwoChannel,
QualityControl

Description QC pipeline and data analysis tools for high-dimensional
Illumina mRNA expression data.

License GPL (>= 2)

NeedsCompilation no

git_url <https://git.bioconductor.org/packages/iCheck>

git_branch RELEASE_3_23

git_last_commit 665af6b

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-09-22

Contents

boxPlots	2
densityPlots	4
genExprSet	6
genSimData.BayesNormal	7
getPCAFunc	9
glmWrapper	10
lkhrrWrapper	13
lmFitPaired	15
lmFitWrapper	18
LumiBatch2Table	21
pca2DPlot	22
pca3DPlot	24
plotCurves	27
plotQCCurves	28
plotSamplep95p05	31
quantilePlot	34
R2PlotFunc	36
scatterPlots	38
sortExpressionSet	41
Index	43

boxPlots	<i>Draw parallel plots for top results in whole-genome-wide analysis</i>
----------	--

Description

Draw scatter plots for top results in whole-genome-wide analysis to test for the association of probes to a continuous-type phenotype variable.

Usage

```
boxPlots(
  resFrame,
  es,
  col.resFrame = c("probeIDs", "stats", "pval", "p.adj"),
  var.pheno = "sex",
  var.probe = "TargetID",
  var.gene = "Symbol",
  var.chr = "Chr",
  nTop = 20,
  myylab = "expression level",
  datExtrFunc = exprs,
  fileFlag = FALSE,
  fileFormat = "ps",
  fileName = "boxPlots.ps")
```

Arguments

<code>resFrame</code>	A data frame stores testing results, which must contain columns that indicate probe id, test statistic, p-value and optionally adjusted p-value.
<code>es</code>	An ExpressionSet object that used to run the whole genome-wide tests.
<code>col.resFrame</code>	A vector of characters indicating column names of <code>resFrame</code> corresponding to probe id, test statistic, p-value and optionally adjusted p-value.
<code>var.pheno</code>	character. the name of continuous-type phenotype variable that is used to test the association of this variable to probes.
<code>var.probe</code>	character. the name of feature variable indicating probe id.
<code>var.gene</code>	character. the name of feature variable indicating gene symbol.
<code>var.chr</code>	character. the name of feature variable indicating chromosome number.
<code>nTop</code>	integer. indicating how many top tests will be used to draw the scatter plot.
<code>myylab</code>	character. indicating y-axis label.
<code>datExtrFunc</code>	name of the function to extract genomic data. For an ExpressionSet object, you should set <code>datExtrFunc=exprs</code> ; for a MethyLumiSet object, you should set <code>datExtrFunc=betas</code> .
<code>fileFlag</code>	logic. indicating if plot should be saved to an external figure file.
<code>fileFormat</code>	character. indicating the figure file type. Possible values are “ps”, “pdf”, or “jpeg”. All other values will produce “png” file.
<code>fileName</code>	character. indicating figure file name (file extension should be specified). For example, you set <code>fileFormat="pdf"</code> , then you can set <code>fileName="test.pdf"</code> , but not <code>fileName="test"</code> .

Value

Value 0 will be returned if no error occurs.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

res.limma = lmFitWrapper(
```

```

es = es.sim,
formula = ~as.factor(memSubj),
pos.var.interest = 1,
pvalAdjMethod = "fdr",
alpha = 0.05,
probeID.var = "probe",
gene.var = "gene",
chr.var = "chr",
verbose = TRUE)

boxPlots(
  resFrame=res.limma$frame,
  es=es.sim,
  col.resFrame = c("probeIDs", "stats", "pval"),
  var.pheno = "memSubj",
  var.probe = "probe",
  var.gene = "gene",
  var.chr = "chr",
  nTop = 20,
  myylab = "expression level",
  datExtrFunc = exprs,
  fileFlag = FALSE,
  fileFormat = "ps",
  fileName = "boxPlots.ps")

```

densityPlots

Draw estimated density plots for all arrays

Description

Draw estimated density plots for all arrays.

Usage

```

densityPlots(
  es,
  requireLog2 = TRUE,
  myxlab = "expression level",
  mymain = "density plots",
  datExtrFunc = exprs,
  fileFlag = FALSE,
  fileFormat = "ps",
  fileName = "densityPlots.ps")

```

Arguments

es	An ExpressionSet object that used to run the whole genome-wide tests.
requireLog2	logic. indicating if log2 transformation is required before estimating densities.

myxlab	character. indicating x-axis label.
mymain	character. indicating title of the plot.
datExtrFunc	name of the function to extract genomic data. For an ExpressionSet object, you should set datExtrFunc=exprs; for a MethyLumiSet object, you should set datExtrFunc=betas.
fileFlag	logic. indicating if plot should be saved to an external figure file.
fileFormat	character. indicating the figure file type. Possible values are “ps”, “pdf”, or “jpeg”. All other values will produce “png” file.
fileName	character. indicating figure file name (file extension should be specified). For example, you set fileFormat=“pdf”, then you can set fileName=“test.pdf”, but not fileName=“test”.

Value

A list object, the i -th element is the object returned by function density for the i -th array.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

densityPlots(
  es = es.sim,
  requireLog2 = FALSE,
  myxlab = "expression level",
  mymain = "density plots",
  datExtrFunc = exprs,
  fileFlag = FALSE,
  fileFormat = "ps",
  fileName = "densityPlots.ps")
```

genExprSet	<i>Generate an ExpressionSet object</i>
------------	---

Description

Generate a simple ExpressionSet object.

Usage

```
genExprSet(  
  ex,  
  pDat,  
  fDat = NULL,  
  annotation = "lumiHumanAll.db")
```

Arguments

ex	A matrix of expression levels. Rows are gene probes and columns are arrays.
pDat	A data frame describing arrays. Rows are arrays and columns are variables describing arrays. The row names of pDat must be the same as the column of ex.
fDat	A data frame describing gene probes. Rows are gene probes and columns are variables describing gene probes. The rownames of fDat must be the same as that of ex.
annotation	character string. Indicating the annotation library (e.g. lumiHumanAll.db for the gene probes.

Value

an ExpressionSet object.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderma <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

genSimData.BayesNormal

Generating simulated data set from conditional normal distributions

Description

Generating simulated data set from conditional normal distributions.

Usage

```
genSimData.BayesNormal(
  nCpGs,
  nCases,
  nControls,
  mu.n = -2,
  mu.c = 2,
  d0 = 20,
  s02 = 0.64,
  s02.c = 1.5,
  testPara = "var",
  outlierFlag = FALSE,
  eps = 0.001,
  applier = lapply)
```

Arguments

nCpGs	integer. Number of genes.
nCases	integer. Number of cases.
nControls	integer. Number of controls.
mu.n	numeric. mean of the conditional normal distribution for controls. See details.
mu.c	numeric. mean of the conditional normal distribution for cases. See details.
d0	integer. degree of freedom for scale-inverse chi squared distribution. See details.
s02	numeric. scaling parameter for scale-inverse chi squared distribution for controls. See details.
s02.c	numeric. scaling parameter for scale-inverse chi squared distribution for cases. See details.
testPara	character string. indicating if the test is for testing equal mean, equal variance, or both.
outlierFlag	logical. indicating if outliers would be generated. If outlierFlag=TRUE, then we followed Phipson and Oshlack's (2014) simulation studies to generate one outlier for each CpG site by replacing the DNA methylation level of one diseased subject by the maximum of the DNA methylation levels of all CpG sites.

eps	numeric. if $ mean0 - mean1 < eps$ then we regard $mean0 = mean1$. Similarly, if $ var0 - var1 < eps$ then we regard $var0 = var1$. $mean0$ and $var0$ are the mean and variance of the chi squared distribution for controls. $mean1$ and $var1$ are the mean and variance of the chi squared distribution for cases.
applier	function name to do apply operation.

Details

Based on Phipson and Oshlack's (2014) simulation algorithm. For each CpG site, variance of the DNA methylation was first sampled from a scaled inverse chi-squared distribution with degree of freedom d_0 and scaling parameter s_0^2 : $\sigma_i^2 \text{ scale} - inv\chi^2(d_0, s_0^2)$. M value for each CpG was then sampled from a normal distribution with mean μ_n and variance equal to the simulated variance σ_i^2 . For cases, the variance was first generated from $\sigma_{i,c}^2 \text{ scale} - inv\chi^2(d_0, s_{0,c}^2)$. M value for each CpG was then sampled from a normal distribution with mean μ_c and variance equal to the simulated variance $\sigma_{i,c}^2$.

Value

An ExpressionSet object. The phenotype data of the ExpressionSet object contains 2 columns: arrayID (array id) and memSubj (subject membership, i.e., case (memSubj=1) or control (memSubj=0)). The feature data of the ExpressionSet object contains 4 elements: probe (probe id), gene (psuedo gene symbol), chr (psuedo chromosome number), and memGenes (indicating if a gene is differentially expressed (when testPara="mean") or indicating if a gene is differentially variable (when testPara="var")).

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

References

Phipson B, Oshlack A. DiffVar: A new method for detecting differential variability with application to methylation in cancer and aging. *Genome Biol* 2014; 15:465

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)
```

getPCAFunc

*Get principal components of arrays***Description**

Get principal components of arrays.

Usage

```
getPCAFunc(es,
            labelVariable = "subjID",
            hybName = "Hybridization_Name",
            requireLog2 = TRUE,
            corFlag = FALSE
          )
```

Arguments

es	An ExpressionSet object
labelVariable	A character string. The name of a phenotype data variable use to label the arrays in the boxplots. By default, labelVariable = "subjID" which is equivalent to labelVariable = "Hybridization_Name".
hybName	character string. indicating the phenotype variable Hybridization_Name.
requireLog2	logical. requiredLog2=TRUE indicates probe expression levels will be log2 transformed. Otherwise, no transformation will be performed.
corFlag	logical. Indicating if correlation matrix (corFlag=TRUE) or covariance (corFlag=FALSE) is used to obtain principal components.

Value

A list with 3 elements:

es.s	An ExpressionSet object with the arrays sorted according to Batch_Run_Date, Chip_Barcode, and Chip_Address
pcs	An object returned by the function prcomp of the R package stats. It contains the following components. sdev (the square roots of the eigenvalues of the covariance/correlation matrix); rotation (a matrix whose columns contain the eigenvectors); x (a matrix whose columns contain principal components); center (the centering used or FALSE); scale (the scale used or FALSE)
requireLog2	logical. The same value as the input requireLog2.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvj@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

pca.obj = getPCAFunc(es = es.sim,
  labelVariable = "subjID",
  hybName = "memSubj",
  requireLog2 = FALSE,
  corFlag = FALSE
)
```

glmWrapper

Perform glm test for all gene probes

Description

Perform glm test for all gene probes.

Usage

```
glmWrapper(es,
  formula = FEV1 ~ xi + age + gender,
  pos.var.interest = 1,
  family = gaussian,
  logit = FALSE,
  pvalAdjMethod = "fdr",
  alpha = 0.05,
  probeID.var = "ProbeID",
  gene.var = "Symbol",
  chr.var = "Chromosome",
  applier = lapply,
  verbose = TRUE)
```

Arguments

es	An LumiBatch object. fData(es) should contains information about probe ID, chromosome number and gene symbol.
formula	An object of class formula. The left handside of ~ is the response variable. Gene probe must be represented by the variable xi. For example, xi~age+gender (gene probe is the response variable); Or FEV1~xi+age+gender (gene probe is the predictor).

pos.var.interest	integer. Indicates which covariate in the right-hand-side of ~ of formula is of the interest. pos.var.interest = 0 means the intercept is of the interest. If the covariate of the interest is an factor or interaction term with more than 2 levels, the smallest p-value will represent the pvalue for the covariate of the interest.
family	By default is gaussian. refer to glm .
logit	logical. Indicate if the gene probes will be logit transformed. For example, for DNA methylation data, one might want to logit transformation for the beta-value ($methyalted/(methyalted + unmethyalted)$).
pvalAdjMethod	One of p-value adjustment methods provided by the R function p.adjust in R package stats: "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none".
alpha	Significance level. A test is claimed to be significant if the adjusted p-value < alpha.
probeID.var	character string. Name of the variable indicating probe ID in feature data set.
gene.var	character string. Name of the variable indicating gene symbol in feature data set.
chr.var	character string. Name of the variable indicating chromosome number in feature data set.
applier	By default, it is lapply. If the library multicore is available, can use mclapply to replace lapply.
verbose	logical. Determine if intermediate output need to be suppressed. By default verbose=TRUE, intermediate output will be printed.

Details

This function applies R function `glm` for each gene probe.

Value

A list with the following elements:

n.sig	Number of significant tests after p-value adjustment.
frame	A data frame containing test results sorted according to the ascending order of unadjusted p-values for the covariate of the interest. The data frame contains 7 columns: probeIDs, geneSymbols (gene symbols of the genes where the probes come from), chr (numbers of chromosomes where the probes locate), stats (z-value), pval (p-values of the tests for the covariate of the interest), p.adj (adjusted p-values), pos (row numbers of the probes in the expression data matrix).
statMat	A matrix containing test statistics for all covariates and for all probes. Rows are probes and columns are covariates. The rows are ordered according to the ascending order of unadjusted p-values for the covariate of the interest.
pvalMat	A matrix containing pvalues for all covariates and for all probes. Rows are probes and columns are covariates. The rows are ordered according to the ascending order of unadjusted p-values for the covariate of the interest.

<code>pval.quantile</code>	Quantiles (minimum, 25 for each covariate including intercept provided in the input argument formula).
<code>frame.unsorted</code>	A data frame containing test results. The data frame contains 7 columns: <code>probeIDs</code> , <code>geneSymbols</code> (gene symbols of the genes where the probes come from), <code>chr</code> (numbers of chromosomes where the probes locate), <code>stats</code> (z-value for the covariate of the interest), <code>pval</code> (p-values of the tests for the covariate of the interest), <code>p.adj</code> (adjusted p-values), <code>pos</code> (row numbers of the probes in the expression data matrix).
<code>statMat.unsorted</code>	A matrix containing test statistics for all covariates and for all probes. Rows are probes and columns are covariates.
<code>pvalMat.unsorted</code>	A matrix containing pvalues for all covariates and for all probes. Rows are probes and columns are covariates.
<code>memGenes</code>	A numeric vector indicating the cluster membership of probes (unsorted). <code>memGenes[i]=1</code> if the <i>i</i> -th probe is significant (adjusted pvalue < alpha) with positive z-value for the covariate of the interest; <code>memGenes[i]=2</code> if the <i>i</i> -th probe is nonsignificant ; <code>memGenes[i]=3</code> if the <i>i</i> -th probe is significant with negative z-value for the covariate of the interest;
<code>memGenes2</code>	A numeric vector indicating the cluster membership of probes (unsorted). <code>memGenes2[i]=1</code> if the <i>i</i> -th probe is significant (adjusted pvalue < alpha). <code>memGenes2[i]=0</code> if the <i>i</i> -th probe is nonsignificant.
<code>mu1</code>	Mean expression levels for arrays for probe cluster 1 (average taking across all probes with <code>memGenes</code> value equal to 1.
<code>mu2</code>	Mean expression levels for arrays for probe cluster 2 (average taking across all probes with <code>memGenes</code> value equal to 2.
<code>mu3</code>	Mean expression levels for arrays for probe cluster 3 (average taking across all probes with <code>memGenes</code> value equal to 3.
<code>resMat</code>	A matrix with $2p$ columns, where p is the number of covariates (including intercept; for a nominal variable with 3 levels say, there were 2 dummy covariates). The first p columns are p-values. The remaining p columns are test statistics.

Note

If the covariate of the interest is a factor or interaction term with more than 2 levels, then the p-value of the likelihood ratio test might be more appropriate than the smallest p-value for the covariate of the interest.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderma <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

res.glm = glmWrapper(
  es = es.sim,
  formula = xi~as.factor(memSubj),
  pos.var.interest = 1,
  family = gaussian,
  logit = FALSE,
  pvalAdjMethod = "fdr",
  alpha = 0.05,
  probeID.var = "probe",
  gene.var = "gene",
  chr.var = "chr",
  applier = lapply,
  verbose = TRUE)
```

lkhRWrapper

Perform glm test for all gene probes

Description

Perform glm test for all gene probes.

Usage

```
lkhRWrapper(es,
  formulaReduced = FEV1 ~ xi + gender,
  formulaFull = FEV1 ~ xi + age + gender,
  family = gaussian,
  logit = FALSE,
  pvalAdjMethod = "fdr",
  alpha = 0.05,
  probeID.var = "ProbeID",
  gene.var = "Symbol",
  chr.var = "Chromosome",
  applier = lapply,
  verbose = TRUE)
```

Arguments

<code>es</code>	An LumiBatch object. <code>fData(es)</code> should contains information about probe ID, chromosome number and gene symbol.
<code>formulaReduced</code>	An object of class formula. Formula for reduced model. The left handside of <code>~</code> is the response variable. Gene probe must be represented by the variable <code>xi</code> . For example, <code>xi~gender</code> (gene probe is the response variable); Or <code>FEV1~xi+gender</code> (gene probe is the predictor).
<code>formulaFull</code>	An object of class formula. Formula for Full model. The left handside of <code>~</code> is the response variable. Gene probe must be represented by the variable <code>xi</code> . For example, <code>xi~age+gender</code> (gene probe is the response variable); Or <code>FEV1~xi+age+gender</code> (gene probe is the predictor).
<code>family</code>	By default is gaussian. refer to glm .
<code>logit</code>	logical. Indicate if the gene probes will be logit transformed. For example, for DNA methylation data, one might want to logit transformation for the beta-value ($methy\!l\!a\!t\!e\!d/(methy\!l\!a\!t\!e\!d + unmethy\!l\!a\!t\!e\!d)$).
<code>pvalAdjMethod</code>	One of p-value adjustment methods provided by the R function <code>p.adjust</code> in R package <code>stats</code> : “holm”, “hochberg”, “hommel”, “bonferroni”, “BH”, “BY”, “fdr”, “none”.
<code>alpha</code>	Significance level. A test is claimed to be significant if the adjusted p-value < alpha.
<code>probeID.var</code>	character string. Name of the variable indicating probe ID in feature data set.
<code>gene.var</code>	character string. Name of the variable indicating gene symbol in feature data set.
<code>chr.var</code>	character string. Name of the variable indicating chromosome number in feature data set.
<code>applier</code>	By default, it is <code>lapply</code> . If the library <code>multicore</code> is available, can use <code>mclapply</code> to replace <code>lapply</code> .
<code>verbose</code>	logical. Determine if intermediate output need to be suppressed. By default <code>verbose=TRUE</code> , intermediate output will be printed.

Details

This function applies R functions `lrtest` in R package `lmtest` and `glm` for each gene probe.

Value

A list with the following elements:

<code>frame</code>	A data frame containing test results sorted according to the ascending order of unadjusted p-values for the covariate of the interest. The data frame contains 8 columns: <code>probeIDs</code> , <code>geneSymbols</code> (gene symbols of the genes where the probes come from), <code>chr</code> (numbers of chromosomes where the probes locate), <code>Chisq</code> (chi square test statistic), <code>Df</code> (degree of freedom of the chisquare test statistic), <code>pval</code> (p-values of the tests for the covariate of the interest), <code>p.adj</code> (adjusted p-values), <code>pos</code> (row numbers of the probes in the expression data matrix). The rows are ordered based on the descending order of chisquare test statistic.
<code>frame.unsorted</code>	A data frame containing test results. unordered frame.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

set.seed(1234567)
es.sim$age = rnorm(ncol(es.sim), mean=50, sd=5)
res.lkh = lkhrWrapper(
  es = es.sim,
  formulaReduced = xi ~ memSubj,
  formulaFull = xi ~ memSubj + age,
  family = gaussian(),
  logit = FALSE,
  pvalAdjMethod = "fdr",
  alpha = 0.05,
  probeID.var = "probe",
  gene.var = "gene",
  chr.var = "chr",
  applier = lapply,
  verbose = TRUE)
```

lmFitPaired

A wrapper function for the function 'lmFit' of the R Bioconductor package 'limma' for paired data

Description

A wrapper function for the function 'lmFit' of the R Bioconductor package 'limma' for paired data.

Usage

```
lmFitPaired(
  esDiff,
  formula = ~1,
  pos.var.interest = 0,
  pvalAdjMethod = "fdr",
```

```
alpha = 0.05,
probeID.var="ProbeID",
gene.var = "Symbol",
chr.var = "Chromosome",
verbose = TRUE)
```

Arguments

<code>esDiff</code>	An LumiBatch object containing log2 difference between cases and controls. <code>fData(esDiff)</code> should contains information about probe ID, chromosome number and gene symbol.
<code>formula</code>	An object of class formula. The intercept measures the effect of treatment. Other covariates measure the effects of their interaction and treatment. The p-values for the intercept will be output. No left handside of <code>~</code> should be specified since the response variable will be the expression level.
<code>pos.var.interest</code>	integer. Indicates which covariate on the right-hand-side of <code>~</code> in formula is the covariate of the interest. By default, it is the intercept <code>pos.var.interest=0</code> .
<code>pvalAdjMethod</code>	One of p-value adjustment methods provided by the R function <code>p.adjust</code> in R package stats: "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none".
<code>alpha</code>	Significance level. A test is claimed to be significant if the adjusted p-value < alpha.
<code>probeID.var</code>	character string. Name of the variable indicating probe ID in feature data set.
<code>gene.var</code>	character string. Name of the variable indicating gene symbol in feature data set.
<code>chr.var</code>	character string. Name of the variable indicating chromosome number in feature data set.
<code>verbose</code>	logical. Determine if intermediate output need to be suppressed. By default <code>verbose=TRUE</code> , intermediate output will be printed.

Details

This is a wrapper function of R Bioconductor functions `lmFit` and `eBayes` for paired data to make it easier to input design and output list of significant results.

Value

A list with the following elements:

<code>n.sig</code>	Number of significant tests after p-value adjustment.
<code>frame</code>	A data frame containing test results sorted according to the ascending order of unadjusted p-values for the intercept. The data frame contains 7 columns: <code>probeIDs</code> , <code>geneSymbols</code> (gene symbols of the genes where the probes come from), <code>chr</code> (numbers of chromosomes where the probes locate), <code>stats</code> (moderated t-statistics for the intercept), <code>pval</code> (p-values of the tests for the intercept), <code>p.adj</code> (adjusted p-values), <code>pos</code> (row numbers of the probes in the expression data matrix).


```

mu.n = -2, mu.c = 2,
d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
outlierFlag = FALSE,
eps = 1.0e-3, applier = lapply)
print(es.sim)

# although the generated data is not from
# paired design, we use it to illustrate the
# usage of the function lmFitPaired

res.limma = lmFitPaired(
  es = es.sim,
  formula = ~as.factor(memSubj),
  pos.var.interest = 0, # the intercept is what we are interested
  pvalAdjMethod = "fdr",
  alpha = 0.05,
  probeID.var = "probe",
  gene.var = "gene",
  chr.var = "chr",
  verbose = TRUE)

```

lmFitWrapper

A wrapper function for the function 'lmFit' of the R Bioconductor package 'limma'

Description

A wrapper function for the function 'lmFit' of the R Bioconductor package 'limma'.

Usage

```

lmFitWrapper(
  es,
  formula = ~as.factor(gender),
  pos.var.interest = 1,
  pvalAdjMethod = "fdr",
  alpha = 0.05,
  probeID.var = "ProbeID",
  gene.var = "Symbol",
  chr.var = "Chromosome",
  verbose = TRUE)

```

Arguments

es	An LumiBatch object. fData(es) should contains information about chromosome number and gene symbol.
formula	An object of class formula. No left handside of ~ should be specified since the response variable will be the expression level.

<code>pos.var.interest</code>	integer. Indicates which covariate on the right-hand-side of <code>~</code> in formula is the covariate of the interest. By default, it is the first covariate <code>pos.var.interest=1</code> .
<code>pvalAdjMethod</code>	One of p-value adjustment methods provided by the R function <code>p.adjust</code> in R package <code>stats</code> : “holm”, “hochberg”, “hommel”, “bonferroni”, “BH”, “BY”, “fdr”, “none”.
<code>alpha</code>	Significance level. A test is claimed to be significant if the adjusted p-value $<$ alpha.
<code>probeID.var</code>	character string. Name of the variable indicating probe ID in feature data set.
<code>gene.var</code>	character string. Name of the variable indicating gene symbol in feature data set.
<code>chr.var</code>	character string. Name of the variable indicating chromosome number in feature data set.
<code>verbose</code>	logical. Determine if intermediate output need to be suppressed. By default <code>verbose=TRUE</code> , intermediate output will be printed.

Details

This is a wrapper function of R Bioconductor functions `lmFit` and `eBayes` to make it easier to input design and output list of significant results.

Value

A list with the following elements:

<code>n.sig</code>	Number of significant tests after p-value adjustment.
<code>frame</code>	A data frame containing test results sorted according to the ascending order of unadjusted p-values for the covariate of the interest. The data frame contains 7 columns: <code>probeIDs</code> , <code>geneSymbols</code> (gene symbols of the genes where the probes come from), <code>chr</code> (numbers of chromosomes where the probes locate), <code>stats</code> (moderated t-statistics for the covariate of interest, i.e. the first covariate), <code>\codepval</code> (p-values of the tests for the covariate of interest, i.e. the first covariate), <code>p.adj</code> (adjusted p-values), <code>pos</code> (row numbers of the probes in the expression data matrix).
<code>statMat</code>	A matrix containing test statistics for all covariates and for all probes. Rows are probes and columns are covariates. The rows are ordered according to the ascending order of unadjusted p-values for the covariate of the interest.
<code>pvalMat</code>	A matrix containing pvalues for all covariates and for all probes. Rows are probes and columns are covariates. The rows are ordered according to the ascending order of unadjusted p-values for the covariate of the interest.
<code>pval.quantile</code>	Quantiles (minimum, 25 for all covariates including intercept provided in the input argument formula.
<code>frame.unsorted</code>	A data frame containing test results. The data frame contains 7 columns: <code>probeIDs</code> , <code>geneSymbols</code> (gene symbols of the genes where the probes come from), <code>chr</code> (numbers of chromosomes where the probes locate), <code>stats</code> (moderated t-statistics for the covariate of the interest), <code>pval</code> (p-values of the tests for the covariate of

	the interest), <code>p.adj</code> (adjusted p-values), <code>pos</code> (row numbers of the probes in the expression data matrix).
<code>statMat.unsorted</code>	A matrix containing test statistics for all covariates and for all probes. Rows are probes and columns are covariates.
<code>pvalMat.unsorted</code>	A matrix containing pvalues for all covariates and for all probes. Rows are probes and columns are covariates.
<code>memGenes</code>	A numeric vector indicating the cluster membership of probes (unsorted). <code>memGenes[i]=1</code> if the <i>i</i> -th probe is significant (adjusted pvalue < alpha) with positive moderated t-statistic; <code>memGenes[i]=2</code> if the <i>i</i> -th probe is nonsignificant ; <code>memGenes[i]=3</code> if the <i>i</i> -th probe is significant with negative moderated t-statistic;
<code>memGenes2</code>	A numeric vector indicating the cluster membership of probes (unsorted). <code>memGenes2[i]=1</code> if the <i>i</i> -th probe is significant (adjusted pvalue < alpha). <code>memGenes2[i]=0</code> if the <i>i</i> -th probe is nonsignificant.
<code>mu1</code>	Mean expression levels for arrays for probe cluster 1 (average taking across all probes with <code>memGenes</code> value equal to 1.
<code>mu2</code>	Mean expression levels for arrays for probe cluster 2 (average taking across all probes with <code>memGenes</code> value equal to 2.
<code>mu3</code>	Mean expression levels for arrays for probe cluster 3 (average taking across all probes with <code>memGenes</code> value equal to 3.
<code>ebFit</code>	object returned by R Bioconductor function <code>eBayes</code> .

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvj@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

res.limma = lmFitWrapper(
  es = es.sim,
  formula = ~as.factor(memSubj),
  pos.var.interest = 1,
  pvalAdjMethod = "fdr",
  alpha = 0.05,
```

```
probeID.var = "probe",
gene.var = "gene",
chr.var = "chr",
verbose = TRUE)
```

LumiBatch2Table	<i>Output slots (exprs, pData, fData) of an LumiBatch object into 3 text files</i>
-----------------	--

Description

Output slots (exprs, pData, fData) of an LumiBatch object into 3 text files.

Usage

```
LumiBatch2Table(
  es,
  probeID.var="ProbeID",
  gene.var="Symbol",
  chr.var="Chromosome",
  sep = ",",
  quote = FALSE,
  filePrefix = "test",
  fileExt = "csv")
```

Arguments

es	An LumiBatch object
probeID.var	character string. Name of the variable indicating probe ID in feature data set.
gene.var	character string. Name of the variable indicating gene symbol in feature data set.
chr.var	character string. Name of the variable indicating chromosome number in feature data set.
sep	Field delimiter for the output text files
quote	logical. Indicating if any character or factor. See also write.table .
filePrefix	Prefix of the names of the output files.
fileExt	File extension of the names of the output files.

Details

Suppose filePrefix="test" and fileExt=".csv". Then, the file names of the 3 output files are: "test_exprs.csv", "test_pDat.csv", and "test_fDat.csv", respectively.

Value

None.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)
```

```
LumiBatch2Table(
  es = es.sim,
  probeID.var="probe",
  gene.var="gene",
  chr.var="chr",
  sep = ",",
  quote = FALSE,
  filePrefix = "test",
  fileExt = "csv")
```

pca2DPlot

Scatter plot of first 2 principal components

Description

Scatter plot of first 2 principal components.

Usage

```
pca2DPlot(pcaObj,
  plot.dim = c(1,2),
  labelVariable = "subjID",
  hybName = "Hybridization_Name",
  outFileName = "test_pca_raw.pdf",
  title = "Scatter plot of pcas",
  plotOutPutFlag = FALSE,
  mar = c(5, 4, 4, 2) + 0.1,
  lwd = 1.5,
  equalRange = TRUE,
  xlab = NULL,
```

```

ylab = NULL,
xlim = NULL,
ylim = NULL,
cex.legend = 1.5,
cex = 1.5,
cex.lab = 1.5,
cex.axis = 1.5,
legendPosition = "topright",
...)

```

Arguments

pcaObj	An object returned by the function <code>pca</code> of the R package <code>pcaMethods</code> .
plot.dim	A vector of 2 positive-integer-value integer specifying which 2 pcas will be plot.
labelVariable	The name of a column of the phenotype data matrix. The elements of the column will replace the column names of the expression data matrix.
hybName	character string. indicating the phenotype variable <code>Hybridization_Name</code> .
outFileName	Name of the figure file to be created.
title	Title of the scatter plot.
plotOutPutFlag	logical. <code>plotOutPutFlag=TRUE</code> indicates the plots will be output to pdf format files. Otherwise, the plots will not be output to external files.
mar	A numerical vector of the form <code>'c(bottom, left, top, right)'</code> which gives the number of lines of margin to be specified on the four sides of the plot. The default is <code>'c(5, 4, 4, 2) + 0.1'</code> . see par .
lwd	The line width, a <code>_positive_</code> number, defaulting to <code>'1'</code> . see par .
equalRange	logical. Indicating if the x-axis and y-axis have the same range.
xlab	Label of x axis.
ylab	Label of y axis.
xlim	Range of x axis.
ylim	Range of y axis.
cex.legend	Font size for legend.
cex	numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. see par .
cex.lab	The magnification to be used for x and y labels relative to the current setting of <code>cex</code> .
cex.axis	The magnification to be used for axis annotation relative to the current setting of <code>cex</code> . see par .
legendPosition	Position of legend. Possible values are "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center".
...	Arguments to be passed to plot .

Value

A matrix of PCA scores. Each column corresponds to a principal component.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

pca.obj = getPcaFunc(es = es.sim,
  labelVariable = "subjID",
  hybName = "memSubj",
  requireLog2 = FALSE,
  corFlag = FALSE
)

pca2DPlot(pcaObj = pca.obj,
  plot.dim = c(1,2),
  labelVariable = "subjID",
  hybName = "memSubj",
  plotOutPutFlag = FALSE,
  cex.legend = 0.5,
  legendPosition = "topright")
```

pca3DPlot

Scatter plot of 3 specified principal components

Description

Scatter plot of 3 specified principal components.

Usage

```
pca3DPlot(pcaObj,
  plot.dim = c(1,2, 3),
  labelVariable = "subjID",
```

```

    hybName = "Hybridization_Name",
    outFileName = "test_pca_raw.pdf",
    title = "Scatter plot of pcas",
    plotOutPutFlag = FALSE,
    mar = c(5, 4, 4, 2) + 0.1,
    lwd = 1.5,
    equalRange = TRUE,
    xlab = NULL,
    ylab = NULL,
    zlab = NULL,
    xlim = NULL,
    ylim = NULL,
    zlim = NULL,
    cex.legend = 1.5,
    cex = 1.5,
    cex.lab = 1.5,
    cex.axis = 1.5,
    legendPosition = "topright",
    ...)

```

Arguments

<code>pcaObj</code>	An object returned by the function <code>pca</code> of the R package <code>pcaMethods</code> .
<code>plot.dim</code>	A vector of 3 positive-integer-value integer specifying which 3 pcas will be plot.
<code>labelVariable</code>	The name of a column of the phenotype data matrix. The elements of the column will replace the column names of the expression data matrix.
<code>hybName</code>	character string. indicating the phenotype variable <code>Hybridization_Name</code> .
<code>outFileName</code>	Name of the figure file to be created.
<code>title</code>	Title of the scatter plot.
<code>plotOutPutFlag</code>	logical. <code>plotOutPutFlag=TRUE</code> indicates the plots will be output to pdf format files. Otherwise, the plots will not be output to external files.
<code>mar</code>	A numerical vector of the form <code>'c(bottom, left, top, right)'</code> which gives the number of lines of margin to be specified on the four sides of the plot. The default is <code>'c(5, 4, 4, 2) + 0.1'</code> . see par .
<code>lwd</code>	The line width, a <code>_positive_</code> number, defaulting to <code>'1'</code> . see par .
<code>equalRange</code>	logical. Indicating if the x-axis and y-axis have the same range.
<code>xlab</code>	Label of x axis.
<code>ylab</code>	Label of y axis.
<code>zlab</code>	Label of z axis.
<code>xlim</code>	Range of x axis.
<code>ylim</code>	Range of y axis.
<code>zlim</code>	Range of z axis.
<code>cex.legend</code>	Font size for legend.

<code>cex</code>	numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. see par .
<code>cex.lab</code>	The magnification to be used for x and y labels relative to the current setting of <code>cex</code> .
<code>cex.axis</code>	The magnification to be used for axis annotation relative to the current setting of <code>cex</code> . see par .
<code>legendPosition</code>	Position of legend. Possible values are “bottomright”, “bottom”, “bottomleft”, “left”, “topleft”, “top”, “topright”, “right” and “center”.
<code>...</code>	Arguments to be passed to plot .

Value

A matrix of PCA scores. Each column corresponds to a principal component.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

pca.obj = getPcaFunc(es = es.sim,
  labelVariable = "subjID",
  hybName = "memSubj",
  requireLog2 = FALSE,
  corFlag = FALSE
)

pca3DPlot(pcaObj = pca.obj,
  plot.dim = c(1,2,3),
  labelVariable = "subjID",
  hybName = "memSubj",
  plotOutPutFlag = FALSE,
  cex.legend = 0.5,
  legendPosition = "topright")
```

plotCurves

Plot trajectories of probe profiles across arrays

Description

Plot trajectories of probe profiles across arrays

Usage

```
plotCurves(
  dat,
  curveNames,
  fileName,
  plotOutPutFlag=FALSE,
  requireLog2 = FALSE,
  cex = 1,
  ylim = NULL,
  xlab = "",
  ylab = "intensity",
  lwd = 3,
  main = "Trajectory plot",
  mar = c(10, 4, 4, 2) + 0.1,
  las = 2,
  cex.axis=1,
  ...)
```

Arguments

dat	Numeric data matrix. Rows are probes; columns are arrays.
curveNames	Probe names.
fileName	file name of output figure.
plotOutPutFlag	logical. plotOutPutFlag=TRUE indicates the plots will be output to pdf format files. Otherwise, the plots will not be output to external files.
requireLog2	logical. requiredLog2=TRUE indicates probe expression levels will be log2 transformed. Otherwise, no transformation will be performed.
cex	numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. see par .
ylim	Range of y axis.
xlab	Label of x axis.
ylab	Label of y axis.
lwd	The line width, a <code>_positive_</code> number, defaulting to '1'. see par .
main	Main title of the plot.

<code>mar</code>	A numerical vector of the form 'c(bottom, left, top, right)' which gives the number of lines of margin to be specified on the four sides of the plot. The default is 'c(5, 4, 4, 2) + 0.1'. see par .
<code>las</code>	'las' numeric in 0,1,2,3; the style of axis labels. 0 - always parallel to the axis, 1 - always horizontal, 2 - always perpendicular to the axis, or 3 - always vertical. see par .
<code>cex.axis</code>	The magnification to be used for axis annotation relative to the current setting of <code>cex</code> . see par .
<code>...</code>	Arguments to be passed to plot .

Value

no return value.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

# plot trajectories of the first 5 genes
plotCurves(
  dat = exprs(es.sim)[1:5,],
  curveNames = featureNames(es.sim)[1:5],
  plotOutPutFlag=FALSE,
  cex = 0.5,
  requireLog2 = FALSE)
```

plotQCCurves

Plot trajectories of specific QC probes (e.g., biotin, cy3_hyb, house-keeping gene probes, low stringency probes, etc.) across arrays

Description

Plot trajectories of specific QC probes (e.g., biotin, cy3_hyb, housekeeping gene probes, low stringency probes, etc.) across arrays

Usage

```
plotQCCurves(
  esQC,
  probes = c("biotin", "cy3_hyb", "housekeeping",
    "low_stringency_hyb", "signal", "p95p05"),
  labelVariable = "subjID",
  hybName = "Hybridization_Name",
  reporterGroupName = "Reporter_Group_Name",
  requireLog2 = TRUE,
  projectName = "test",
  plotOutPutFlag = FALSE,
  cex = 1,
  ylim = NULL,
  xlab = "",
  ylab = "intensity",
  lwd = 3,
  mar = c(10, 4, 4, 2) + 0.1,
  las = 2,
  cex.axis = 1,
  sortFlag = TRUE,
  varSort = c("Batch_Run_Date", "Chip_Barcode", "Chip_Address"),
  timeFormat = c("%m/%d/%Y", NA, NA),
  ...)
```

Arguments

esQC	ExpressionSet object of QC probe profiles. fData(esQC) should contains the variable Reporter_Group_Name.
probes	A character vectors of QC probe names. By default, it includes the following probe names “biotin”, “cy3_hyb”, “housekeeping”, “low_stringency_hyb”, “signal”, “p95p05”. For “signal”, trajectories of 5th, 25th, 50th, 75th, and 95th percentiles of the expression levels of all QC probes will be plotted. For “p95p05”, the trajectory of the ratio of 95th percentile to 5th percentile of the expression levels of all QC probes will be plotted.
labelVariable	A character string. The name of a phenotype data variable use to label the arrays in the boxplots. By default, labelVariable = “subjID” which is equivalent to labelVariable = “Hybridization_Name”.
hybName	character string. indicating the phenotype variable Hybridization_Name.
reporterGroupName	character string. indicating feature variable Reporter_Group_Name (QC probe’s name).

requireLog2	logical. requiredLog2=TRUE indicates probe expression levels will be log2 transformed. Otherwise, no transformation will be performed.
projectName	A character string. Name of the project. The plots will be saved as pdf format files, the names of which have the format projectName_probeName_traj_plot.pdf.
plotOutPutFlag	logical. plotOutPutFlag=TRUE indicates the plots will be output to pdf format files. Otherwise, the plots will not be output to external files.
cex	numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. see par .
ylim	Range of y axis.
xlab	Label of x axis.
ylab	Label of y axis.
lwd	The line width, a <code>_positive_</code> number, defaulting to '1'. see par .
mar	A numerical vector of the form 'c(bottom, left, top, right)' which gives the number of lines of margin to be specified on the four sides of the plot. The default is 'c(5, 4, 4, 2) + 0.1'. see par .
las	'las' numeric in 0,1,2,3; the style of axis labels. 0 - always parallel to the axis, 1 - always horizontal, 2 - always perpendicular to the axis, or 3 - always vertical. see par .
cex.axis	The magnification to be used for axis annotation relative to the current setting of cex. see par .
sortFlag	logical. Indicates if arrays need to be sorted according to Batch_Run_Date, Chip_Barcode, and Chip_Address.
varSort	A vector of phenotype variable names to be used to sort the samples of es.
timeFormat	A vector of time format for the possible time variables in varSort. The length of timeFormat should be the same as that of varSort. For non-time variable, the corresponding time format should be set to be equal to NA.
...	Arguments to be passed to plot .

Value

no return value.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
esQC.sim = genSimData.BayesNormal(nCpGs = 10,
  nCases = 20, nControls = 20,
```

```

mu.n = -2, mu.c = 2,
d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
outlierFlag = FALSE,
eps = 1.0e-3, applier = lapply)

print(esQC.sim)

fDat = fData(esQC.sim)
esQC.sim$Hybridization_Name = sampleNames(esQC.sim)
fDat$Reporter_Group_Name = c( rep("biotin", 5),
  rep("housekeeping", 5))
fData(esQC.sim)=fDat

# plot trajectories of the QC probes
plotQCCurves(
  esQC = esQC.sim,
  probes = c("biotin", "housekeeping"),
  labelVariable = "subjID",
  hybName = "Hybridization_Name",
  reporterGroupName = "Reporter_Group_Name",
  requireLog2 = FALSE,
  plotOutPutFlag = FALSE,
  sortFlag = FALSE)

```

plotSamplep95p05

Plot trajectories of the ratio of 95th percentile to 5th percentile of sample probe profiles across arrays

Description

Plot trajectories of the ratio of 95th percentile to 5th percentile of sample probe profiles across arrays.

Usage

```

plotSamplep95p05(
  es,
  labelVariable = "subjID",
  hybName = "Hybridization_Name",
  requireLog2 = FALSE,
  projectName = "test",
  plotOutPutFlag = FALSE,
  cex = 1,
  ylim = NULL,
  xlab = "",
  ylab = "",
  lwd = 1.5,
  mar = c(10, 4, 4, 2) + 0.1,

```

```

las = 2,
cex.axis=1.5,
title = "Trajectory of p95/p05",
cex.legend = 1.5,
cex.lab = 1.5,
legendPosition = "topright",
cut1 = 10,
cut2 = 6,
sortFlag = TRUE,
varSort = c("Batch_Run_Date", "Chip_Barcode", "Chip_Address"),
timeFormat = c("%m/%d/%Y", NA, NA),
verbose = FALSE,
...)

```

Arguments

<code>es</code>	ExpressionSet object of Sample probe profiles.
<code>labelVariable</code>	A character string. The name of a phenotype data variable use to label the arrays in the boxplots. By default, <code>labelVariable = "subjID"</code> which is equivalent to <code>labelVariable = "Hybridization_Name"</code> .
<code>hybName</code>	character string. indicating the phenotype variable <code>Hybridization_Name</code> .
<code>requireLog2</code>	logical. <code>requiredLog2=TRUE</code> indicates probe expression levels will be log2 transformed. Otherwise, no transformation will be performed.
<code>projectName</code>	A character string. Name of the project. The plots will be saved as pdf format files, the names of which have the format <code>projectName_probeName_traj_plot.pdf</code> .
<code>plotOutPutFlag</code>	logical. <code>plotOutPutFlag=TRUE</code> indicates the plots will be output to pdf format files. Otherwise, the plots will not be output to external files.
<code>cex</code>	numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. see par .
<code>ylim</code>	Range of y axis.
<code>xlab</code>	Label of x axis.
<code>ylab</code>	Label of y axis.
<code>lwd</code>	The line width, a <code>_positive_</code> number, defaulting to '1'. see par .
<code>mar</code>	A numerical vector of the form ' <code>c(bottom, left, top, right)</code> ' which gives the number of lines of margin to be specified on the four sides of the plot. The default is ' <code>c(5, 4, 4, 2) + 0.1</code> '. see par .
<code>las</code>	'las' numeric in 0,1,2,3; the style of axis labels. 0 - always parallel to the axis, 1 - always horizontal, 2 - always perpendicular to the axis, or 3 - always vertical. see par .
<code>cex.axis</code>	The magnification to be used for axis annotation relative to the current setting of <code>cex</code> . see par .
<code>title</code>	Figure title.
<code>cex.legend</code>	Font size of legend text.

cex.lab	The magnification to be used for x and y labels relative to the current setting of cex.
legendPosition	Position of legend. Possible values are “bottomright”, “bottom”, “bottomleft”, “left”, “topleft”, “top”, “topright”, “right” and “center”.
cut1	second horizontal line setting the cutoff for the ratio p95/p05. A ratio above this line indicates the corresponding array is good.
cut2	second horizontal line setting the cutoff for the ratio p95/p05. A ratio below this line indicates the corresponding array is bad.
sortFlag	logical. Indicates if arrays need to be sorted according to Batch_Run_Date, Chip_Barcode, and Chip_Address.
varSort	A vector of phenotype variable names to be used to sort the samples of es.
timeFormat	A vector of time format for the possible time variables in varSort. The length of timeFormat should be the same as that of varSort. For non-time variable, the corresponding time format should be set to be equal to NA. The details of the time format for time variable can be found in the R function strptime .
verbose	logical. Determine if intermediate output need to be suppressed. By default verbose=FALSE, intermediate output will not be printed.
...	Arguments to be passed to plot .

Details

The trajectory of the ratio of 95 to 5

Value

A list of 2 elements. The first element is the 2 x n matrix, where n is the number of arrays. The first row of the matrix is the 5-th percentile and the second row of the matrix is the 95-th percentile.

The second element is the ratio of the 95-th percentile to the 5-th percentile.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)
```

```

es.sim$Batch_Run_Date = 1:ncol(es.sim)
es.sim$Chip_Barcode = 1:ncol(es.sim)
es.sim$Chip_Address = 1:ncol(es.sim)

plotSamplep95p05(
  es = es.sim,
  labelVariable = "subjID",
  hybName = "memSubj",
  requireLog2 = FALSE,
  projectName = "test",
  plotOutPutFlag = FALSE,
  title = "Trajectory of p95/p05",
  cex.legend = 0.5,
  legendPosition = "topright",
  sortFlag = TRUE,
  varSort = c("Batch_Run_Date", "Chip_Barcode", "Chip_Address"),
  timeFormat = c("%m/%d/%Y", NA, NA),
  verbose = FALSE)

```

quantilePlot

Plot trajectories of quantiles across arrays

Description

Plot trajectories of quantiles across arrays.

Usage

```

quantilePlot(
  dat,
  fileName,
  probs = c(0, 0.05, 0.25, 0.5, 0.75, 0.95, 1),
  plotOutPutFlag = FALSE,
  requireLog2 = FALSE,
  sortFlag = TRUE,
  cex = 1,
  ylim = NULL,
  xlab = "",
  ylab = "intensity",
  lwd = 3,
  main = "Trajectory plot of quantiles",
  mar = c(15, 4, 4, 2) + 0.1,
  las = 2,
  cex.axis = 1)

```

Arguments

<code>dat</code>	Expression data. Rows are gene probes; columns are arrays.
<code>fileName</code>	File name of output figure.
<code>probs</code>	quantiles (any real values between the interval $[0, 1]$).
<code>plotOutPutFlag</code>	logical. <code>plotOutPutFlag=TRUE</code> indicates the plots will be output to pdf format files. Otherwise, the plots will not be output to external files.
<code>requireLog2</code>	logical. <code>requiredLog2=TRUE</code> indicates probe expression levels will be log2 transformed. Otherwise, no transformation will be performed.
<code>sortFlag</code>	logical. <code>sortFlag=TRUE</code> indicates arrays will be sorted by the ascending order of MAD (median absolute deviation)
<code>cex</code>	numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. see par .
<code>ylim</code>	Range of y axis.
<code>xlab</code>	Label of x axis.
<code>ylab</code>	Label of y axis.
<code>lwd</code>	The line width, a <code>_positive_</code> number, defaulting to '1'. see par .
<code>main</code>	Character string. main title of the plot.
<code>mar</code>	A numerical vector of the form 'c(bottom, left, top, right)' which gives the number of lines of margin to be specified on the four sides of the plot. The default is 'c(5, 4, 4, 2) + 0.1'. see par .
<code>las</code>	'las' numeric in 0,1,2,3; the style of axis labels. 0 - always parallel to the axis, 1 - always horizontal, 2 - always perpendicular to the axis, or 3 - always vertical. see par .
<code>cex.axis</code>	The magnification to be used for axis annotation relative to the current setting of <code>cex</code> . see par .

Value

The quantile matrix with row quantiles and column array.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderma <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
```

```

    d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
    outlierFlag = FALSE,
    eps = 1.0e-3, applier = lapply)
print(es.sim)

png(file="qplot.png")

quantilePlot(
  dat = exprs(es.sim),
  probs = c(0, 0.05, 0.25, 0.5, 0.75, 0.95, 1),
  plotOutPutFlag = FALSE,
  requireLog2 = FALSE,
  sortFlag = TRUE)

dev.off()

```

R2PlotFunc

Draw heatmap of square of correlations among arrays

Description

Draw heatmap of square of correlations among arrays.

Usage

```

R2PlotFunc(
  es,
  hybName = "Hybridization_Name",
  arrayType = c("all", "replicates", "GC"),
  GCid = c("128115", "Hela", "Brain"),
  probs = seq(0, 1, 0.25),
  col = gplots::greenred(75),
  labelVariable = "subjID",
  outFileName = "test_R2_raw.pdf",
  title = "Raw Data R^2 Plot",
  requireLog2 = FALSE,
  plotOutPutFlag = FALSE,
  las = 2,
  keysize = 1,
  margins = c(10, 10),
  sortFlag = TRUE,
  varSort=c("Batch_Run_Date", "Chip_Barcode", "Chip_Address"),
  timeFormat=c("%m/%d/%Y", NA, NA),
  ...)

```

Arguments

<code>es</code>	ExpressionSet object of QC probe profiles.
<code>hybName</code>	character string. indicating the phenotype variable Hybridization_Name.
<code>arrayType</code>	A character string indicating if the correlations are calculated based on all arrays, arrays with replicates, or genetic control arrays.
<code>GCid</code>	A vector of character string. symbols for genetic control samples. The symbols can be more than one.
<code>probs</code>	A vector of probabilities specify the quantiles of correlations to be output.
<code>col</code>	colors used for the image. see the function heatmap.2 in R package gplots.
<code>labelVariable</code>	A character string indicating how to label the arrays.
<code>outFileName</code>	A character string. The name of output file.
<code>title</code>	Title of the plot.
<code>requireLog2</code>	logical. <code>requiredLog2=TRUE</code> indicates probe expression levels will be log2 transformed. Otherwise, no transformation will be performed.
<code>plotOutPutFlag</code>	logical. <code>plotOutPutFlag=TRUE</code> indicates the plots will be output to pdf format files. Otherwise, the plots will not be output to external files.
<code>las</code>	'las' numeric in 0,1,2,3; the style of axis labels. 0 - always parallel to the axis, 1 - always horizontal, 2 - always perpendicular to the axis, or 3 - always vertical. see par .
<code>keysize</code>	numeric value indicating the size of the key. see the function heatmap.2 in R package gplots.
<code>margins</code>	numeric vector of length 2 containing the margins. see the function heatmap.2 in R package gplots.
<code>sortFlag</code>	logical. Indicates if arrays need to be sorted according to Batch_Run_Date, Chip_Barcode, and Chip_Address.
<code>varSort</code>	A vector of phenotype variable names to be used to sort the samples of es.
<code>timeFormat</code>	A vector of time format for the possible time variables in varSort. The length of timeFormat should be the same as that of varSort. For non-time variable, the corresponding time format should be set to be equal to NA. The details of the time format for time variable can be found in the R function strptime .
<code>...</code>	Arguments to be passed to heatmap.2 .

Value

A list with 3 elments. The first element R2Mat is the matrix of squared correlation. The second element R2vec is the vector of the upper triangle of the matrix of squared correlation (diagonal elements are excluded). The third element R2vec.within.req is the vector of within-replicate R^2 , that is, any element in R2vec.within.req is the squared correlation coefficient between two arrays/replicates for a subject.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

es.sim$Batch_Run_Date = 1:ncol(es.sim)
es.sim$Chip_Barcode = 1:ncol(es.sim)
es.sim$Chip_Address = 1:ncol(es.sim)

# draw heatmap for the first 5 subjects
png(file="r2plot.png")
R2PlotFunc(
  es = es.sim[, 1:5],
  hybName = "memSubj",
  arrayType = c("all", "replicates", "GC"),
  GCid = c("128115", "Hela", "Brain"),
  probs = seq(0, 1, 0.25),
  col = gplots::greenred(75),
  labelVariable = "subjID",
  outFileName = "test_R2_raw.pdf",
  title = "Raw Data R^2 Plot",
  requireLog2 = FALSE,
  plotOutPutFlag = FALSE,
  las = 2,
  keysize = 1,
  margins = c(10, 10),
  sortFlag = TRUE,
  varSort=c("Batch_Run_Date", "Chip_Barcode", "Chip_Address"),
  timeFormat=c("%m/%d/%Y", NA, NA))
dev.off()
```

scatterPlots

Draw scatter plots for top results in whole-genome-wide analysis

Description

Draw scatter plots for top results in whole-genome-wide analysis to test for the association of probes to a continuous-type phenotype variable.

Usage

```
scatterPlots(
```

```

resFrame,
es,
col.resFrame = c("probeIDs", "stats", "pval", "p.adj"),
var.pheno = "bmi",
outcomeFlag = FALSE,
fitLineFlag = TRUE,
var.probe = "TargetID",
var.gene = "Symbol",
var.chr = "Chr",
nTop = 20,
myylab = "expression level",
datExtrFunc = exprs,
fileFlag = FALSE,
fileFormat = "ps",
fileName = "scatterPlots.ps")

```

Arguments

<code>resFrame</code>	A data frame stores testing results, which must contain columns that indicate probe id, test statistic, p-value and optionally adjusted p-value.
<code>es</code>	An ExpressionSet object that used to run the whole genome-wide tests.
<code>col.resFrame</code>	A vector of characters indicating column names of <code>resFrame</code> corresponding to probe id, test statistic, p-value and optionally adjusted p-value.
<code>var.pheno</code>	character. the name of continuous-type phenotype variable that is used to test the association of this variable to probes.
<code>outcomeFlag</code>	logic. indicating if <code>var.pheno</code> is the outcome variable in regression analysis.
<code>fitLineFlag</code>	logic. indicating if a fitted line $y = a + bx$ should be plotted. If <code>outcomeFlag=TRUE</code> , then y is <code>var.pheno</code> and x is the top probe. If <code>outcomeFlag=FALSE</code> , then y is the top probe and x is <code>var.pheno</code> .
<code>var.probe</code>	character. the name of feature variable indicating probe id.
<code>var.gene</code>	character. the name of feature variable indicating gene symbol.
<code>var.chr</code>	character. the name of feature variable indicating chromosome number.
<code>nTop</code>	integer. indicating how many top tests will be used to draw the scatter plot.
<code>myylab</code>	character. indicating y-axis label.
<code>datExtrFunc</code>	name of the function to extract genomic data. For an ExpressionSet object, you should set <code>datExtrFunc=exprs</code> ; for a MethyLumiSet object, you should set <code>datExtrFunc=betas</code> .
<code>fileFlag</code>	logic. indicating if plot should be saved to an external figure file.
<code>fileFormat</code>	character. indicating the figure file type. Possible values are "ps", "pdf", or "jpeg". All other values will produce "png" file.
<code>fileName</code>	character. indicating figure file name (file extension should be specified). For example, you set <code>fileFormat="pdf"</code> , then you can set <code>fileName="test.pdf"</code> , but not <code>fileName="test"</code> .

Value

Value 0 will be returned if no error occurs.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

# generate phenotype age
es.sim$age = rnorm(ncol(es.sim), mean=50, sd=5)

res.limma = lmFitWrapper(
  es = es.sim,
  formula = ~age,
  pos.var.interest = 1,
  pvalAdjMethod = "fdr",
  alpha = 0.05,
  probeID.var = "probe",
  gene.var = "gene",
  chr.var = "chr",
  verbose = TRUE)

scatterPlots(
  resFrame=res.limma$frame,
  es=es.sim,
  col.resFrame = c("probeIDs", "stats", "pval"),
  var.pheno = "age",
  outcomeFlag = FALSE,
  fitLineFlag = TRUE,
  var.probe = "probe",
  var.gene = "gene",
  var.chr = "chr",
  nTop = 20,
  myylab = "expression level",
  datExtrFunc = exprs,
  fileFlag = FALSE,
  fileFormat = "ps",
  fileName = "scatterPlots.ps")
```

sortExpressionSet	<i>Sort the order of samples for an ExpressionSet object</i>
-------------------	--

Description

Sort the order of samples for an ExpressionSet object.

Usage

```
sortExpressionSet(
  es,
  varSort = c("Batch_Run_Date", "Chip_Barcode", "Chip_Address"),
  timeFormat = c("%m/%d/%Y", NA, NA)
)
```

Arguments

es	An ExpressionSet.
varSort	A vector of phenotype variable names to be used to sort the samples of es.
timeFormat	A vector of time format for the possible time variables in varSort. The length of timeFormat should be the same as that of varSort. For non-time variable, the corresponding time format should be set to be equal to NA. Please refer to function strptime of the base package.

Value

An ExpressionSet object with samples sorted based on the variables indicated in varSort.

Author(s)

Weiliang Qiu <stwxq@channing.harvard.edu>, Brandon Guo <brandowonder@gmail.com>, Christopher Anderson <christopheranderson84@gmail.com>, Barbara Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
  nCases = 20, nControls = 20,
  mu.n = -2, mu.c = 2,
  d0 = 20, s02 = 0.64, s02.c = 1.5, testPara = "var",
  outlierFlag = FALSE,
  eps = 1.0e-3, applier = lapply)
print(es.sim)

es.sim$Batch_Run_Date = 1:ncol(es.sim)
es.sim$Chip_Barcode = 1:ncol(es.sim)
```

```
es.sim$Chip_Address = 1:ncol(es.sim)

es.sim2 = sortExpressionSet(
  es = es.sim,
  varSort = c("Batch_Run_Date", "Chip_Barcode", "Chip_Address"),
  timeFormat = c("%m/%d/%Y", NA, NA)
)
```

Index

* **methods**

- genExprSet, [6](#)
- getPCAFunc, [9](#)
- glmWrapper, [10](#)
- lkhrrWrapper, [13](#)
- lmFitPaired, [15](#)
- lmFitWrapper, [18](#)
- LumiBatch2Table, [21](#)
- pca2DPlot, [22](#)
- pca3DPlot, [24](#)
- plotCurves, [27](#)
- plotQCCurves, [28](#)
- plotSamplep95p05, [31](#)
- quantilePlot, [34](#)
- R2PlotFunc, [36](#)
- sortExpressionSet, [41](#)

* **method**

- boxPlots, [2](#)
- densityPlots, [4](#)
- genSimData.BayesNormal, [7](#)
- scatterPlots, [38](#)

boxPlots, [2](#)

densityPlots, [4](#)

genExprSet, [6](#)
genSimData.BayesNormal, [7](#)
getPCAFunc, [9](#)
glm, [11](#), [14](#)
glmWrapper, [10](#)

heatmap.2, [37](#)

lkhrrWrapper, [13](#)
lmFitPaired, [15](#)
lmFitWrapper, [18](#)
LumiBatch2Table, [21](#)

par, [23](#), [25–28](#), [30](#), [32](#), [35](#), [37](#)
pca2DPlot, [22](#)

pca3DPlot, [24](#)
plot, [23](#), [26](#), [28](#), [30](#), [33](#)
plotCurves, [27](#)
plotQCCurves, [28](#)
plotSamplep95p05, [31](#)

quantilePlot, [34](#)

R2PlotFunc, [36](#)

scatterPlots, [38](#)
sortExpressionSet, [41](#)
strptime, [33](#), [37](#), [41](#)

write.table, [21](#)