Package ‘iCheck’

March 28, 2017

Type Package

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

Version 1.4.0

Date 2016-10-16

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

Suggests 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

R topics documented:

    genExprSet .................................................. 2
    genSimData.BayesNormal ................................. 3
    getPCAFunc .................................................. 4
    glmWrapper .................................................. 6
    lkhrWrapper ................................................. 9
    lmFitPaired .................................................. 11
    lmFitWrapper ................................................. 13
    LumiBatch2Table .......................................... 16
genExprSet

Description

Generate a simple ExpressionSet object.

Usage

```r
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 Klanderman <BKLANDERMAN@partners.org>, Vincent Carey <stvjc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>
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.
**getPCAFunc**

Get principal components of arrays.

**Details**

Based on Phipson and Oshlack’s (2014) simulation algorithm. For each CpG site, variance of the DNA methylation was first sampled from an scaled inverse chi-squared distribution with degree of freedom $d_0$ and scaling parameter $s_0^2$: $\sigma^2_i \sim \text{scale} - \text{inv}\chi^2(d_0, s_0^2)$. M value for each CpG was then sampled from a normal distribution with mean $\mu_n$ and variance equal to the simulated variance $\sigma^2_i$. For cases, the variance was first generated from $\sigma^2_{i,c} \sim \text{scale} - \text{inv}\chi^2(d_0, s_0^2, c)$. M value for each CpG was then sampled from a normal distribution with mean $\mu_c$ and variance equal to the simulated variance $\sigma^2_{i,c}$.

**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**


**Examples**

```r
# 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 used 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 <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,
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-size 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 (methylated/(methylated + unmethylated)).
- `alpha` Significance level. A test is claimed to be significant if the adjusted p-value < alpha.
glmWrapper

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 lappy.
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 p-values 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.
pval.quantile Quantiles (minimum, 25 for each covariate including intercept provided in the input argument formula.
frame.unsorted A data frame containing test results. 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 for the covariate of the interest), 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.unsorted A matrix containing test statistics for all covariates and for all probes. Rows are probes and columns are covariates.
pvalMat.unsorted A matrix containing p-values for all covariates and for all probes. Rows are probes and columns are covariates.
memGenes A numeric vector indicating the cluster membership of probes (unsorted). memGenes[i]=1 if the i-th probe is significant (adjusted pvalue < alpha) with positive z-value
glmWrapper

for the covariate of the interest; memGenes[i]=2 if the i-th probe is nonsignificant; memGenes[i]=3 if the i-th probe is significant with negative z-value for the covariate of the interest;

memGenes2
A numeric vector indicating the cluster membership of probes (unsorted). memGenes2[i]=1 if the i-th probe is significant (adjusted pvalue < alpha). memGenes2[i]=0 if the i-th probe is nonsignificant.

mu1
Mean expression levels for arrays for probe cluster 1 (average taking across all probes with memGenes value equal to 1).

mu2
Mean expression levels for arrays for probe cluster 2 (average taking across all probes with memGenes value equal to 2).

mu3
Mean expression levels for arrays for probe cluster 3 (average taking across all probes with memGenes value equal to 3).

resMat
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 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.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)
Perform glm test for all gene probes.

**Usage**

```r
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**

- **es**: An LumiBatch object. `fData(es)` should contain information about probe ID, chromosome number, and gene symbol.
- **formulaReduced**: An object of class `formula`. Formula for the reduced model. The left hand side of `~` is the response variable. Gene probe must be represented by the variable `xi`. For example, `xi~gender` (gene probe is the response variable); or `FEV1~xi+gender` (gene probe is the predictor).
- **formulaFull**: An object of class `formula`. Formula for the full model. The left hand side 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).
- **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 (`methylated/(methylated + unmethylated)`).
- **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 functions `lrtest` in R package `lmtest` and `glm` for each gene probe.

Value

A list with the following elements:

- `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 8 columns: `probeIDs`, `geneSymbols` (gene symbols of the genes where the probes come from), `chr` (numbers of chromosomes where the probes locate), `Chisq` (chi square test statistic), `Df` (degree of freedom of the chisquare test statistic), `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). The rows are ordered based on the descending order of chisquare test statistic.

- `frame.unsorted`: 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

```r
# 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",
)```
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

esDiff An LumiBatch object containing log2 difference between cases and controls. `fData(esDiff)` should contains information about probe ID, chromosome number and gene symbol.

formula 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 ~ should be specified since the response variable will be the expression level.

pos.var.interest integer. Indicates which covariate on the right-hand-side of ~ in formula is the covariate of the interest. By default, it is the intercept pos.var.interest=0.

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.

verbose logical. Determine if intermediate output need to be suppressed. By default verbose=TRUE, 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:

- **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 intercept. 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` (moderated t-statistics for the intercept), `pval` (p-values of the tests for the intercept), `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 intercept.

- **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 intercept.

- **pval.quantile**
  - Quantiles (minimum, 25 for all covariates including intercept provided in the input argument `formula`).

- **frame.unsorted**
  - A data frame containing test results. 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` (moderated t-statistics for the intercept), `pval` (p-values of the tests for the intercept), `p.adj` (adjusted p-values), `pos` (row numbers of the probes in the expression data matrix).

- **statMat.unsorted**
  - A matrix containing test statistics for all covariates and for all probes. Rows are probes and columns are covariates.

- **pvalMat.unsorted**
  - A matrix containing pvalues for all covariates and for all probes. Rows are probes and columns are covariates.

- **memGenes**
  - A numeric vector indicating the cluster membership of probes (unsorted). `memGenes[i]=1` if the i-th probe is significant (adjusted pvalue < alpha) with positive moderated t-statistic; `memGenes[i]=2` if the i-th probe is nonsignificant; `memGenes[i]=3` if the i-th probe is significant with negative moderated t-statistic;

- **memGenes2**
  - A numeric vector indicating the cluster membership of probes (unsorted). `memGenes2[i]=1` if the i-th probe is significant (adjusted pvalue < alpha). `memGenes2[i]=0` if the i-th probe is nonsignificant.

- **mu1**
  - Mean expression levels for arrays for probe cluster 1 (average taking across all probes with `memGenes` value equal to 1).

- **mu2**
  - Mean expression levels for arrays for probe cluster 2 (average taking across all probes with `memGenes` value equal to 2).

- **mu3**
  - Mean expression levels for arrays for probe cluster 3 (average taking across all probes with `memGenes` value equal to 3).

- **ebFit**
  - object returned by R Bioconductor function `eBayes`. 
lmFitWrapper

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)

# 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 contain information about chromosome number and gene symbol.

**formula**
An object of class `formula`. No left hand-side of ~ should be specified since the response variable will be the expression level.

**pos.var.interest**
integer. Indicates which covariate on the right-hand-side of ~ in `formula` is the covariate of the interest. By default, it is the first covariate `pos.var.interest=1`.

**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.

**verbose**
logical. Determine if intermediate output need to be suppressed. By default `verbose=TRUE`, 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:

**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 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` (moderated t-statistics for the covariate of interest, i.e. the first covariate), `pcodepval` (p-values of the tests for the covariate of interest, i.e. the first covariate), `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 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 interest.

**pval.quantile**
Quantiles (minimum, 25 for all covariates including intercept provided in the input argument `formula`).
frame.unsorted
A data frame containing test results. 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 (moderated t-statistics for the covariate of the interest), 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.unsorted
A matrix containing test statistics for all covariates and for all probes. Rows are probes and columns are covariates.

pvalMat.unsorted
A matrix containing pvalues for all covariates and for all probes. Rows are probes and columns are covariates.

memGenes
A numeric vector indicating the cluster membership of probes (unsorted). memGenes[i]=1 if the i-th probe is significant (adjusted pvalue < alpha) with positive moderated t-statistic; memGenes[i]=2 if the i-th probe is nonsignificant; memGenes[i]=3 if the i-th probe is significant with negative moderated t-statistic;

memGenes2
A numeric vector indicating the cluster membership of probes (unsorted). memGenes2[i]=1 if the i-th probe is significant (adjusted pvalue < alpha). memGenes2[i]=0 if the i-th probe is nonsignificant.

mu1
Mean expression levels for arrays for probe cluster 1 (average taking across all probes with memGenes value equal to 1.

mu2
Mean expression levels for arrays for probe cluster 2 (average taking across all probes with memGenes value equal to 2.

mu3
Mean expression levels for arrays for probe cluster 3 (average taking across all probes with memGenes value equal to 3.

ebFit
object returned by R Bioconductor function eBayes.

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 <stjc@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",
)
LumiBatch2Table

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

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.
pca2DPlot

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,
xislab = 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 pca of the R package pcaMethods.
- 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 Hybridization_Name.
- outFileName: Name of the figure file to be created.
- title: Title of the scatter plot.
- plotOutPutFlag: logical. plotOutPutFlag=TRUE 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 ’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.
- lwd: The line width, a _positive_ number, defaulting to ’1’. 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 cex.
- cex.axis: The magnification to be used for axis annotation relative to the current setting of cex.
- 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**

```r
# 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)
```

**Description**

Scatter plot of 3 specified principal components.

**Usage**

```r
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 = 0.5,
   legendPosition = "topright")
```
pca3DPlot

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 pca of the R package pcaMethods.
plot.dim A vector of 3 positive-integer-value integer specifying which 3 pcs 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 Hybridization_Name.
outFileName Name of the figure file to be created.
title Title of the scatter plot.
plotOutPutFlag logical. plotOutPutFlag=TRUE 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 `c(bottom, left, top, right)` which gives the num-
ber 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.
lwd The line width, a _positive_ number, defaulting to ‘1’. 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.
zlab Label of z axis.
xlim Range of x axis.
ylim Range of y axis.
zlim Range of z 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
cex.
cex.axis The magnification to be used for axis annotation relative to the current setting of
cex.
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.
plotCurves

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

```r
# 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

```r
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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dat</td>
<td>Numeric data matrix. Rows are probes; columns are arrays.</td>
</tr>
<tr>
<td>curveNames</td>
<td>Probe names.</td>
</tr>
<tr>
<td>fileName</td>
<td>File name of output figure.</td>
</tr>
<tr>
<td>plotOutPutFlag</td>
<td>Logical. plotOutPutFlag=TRUE indicates the plots will be output to pdf format files. Otherwise, the plots will not be output to external files.</td>
</tr>
<tr>
<td>requireLog2</td>
<td>Logical. requireLog2=TRUE indicates probe expression levels will be log2 transformed. Otherwise, no transformation will be performed.</td>
</tr>
<tr>
<td>cex</td>
<td>Numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. See par.</td>
</tr>
<tr>
<td>ylim</td>
<td>Range of y axis.</td>
</tr>
<tr>
<td>xlab</td>
<td>Label of x axis.</td>
</tr>
<tr>
<td>ylab</td>
<td>Label of y axis.</td>
</tr>
<tr>
<td>lwd</td>
<td>The line width, a positive number, defaulting to '1'. See par.</td>
</tr>
<tr>
<td>main</td>
<td>Main title of the plot.</td>
</tr>
<tr>
<td>mar</td>
<td>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.</td>
</tr>
<tr>
<td>las</td>
<td>'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.</td>
</tr>
<tr>
<td>cex.axis</td>
<td>The magnification to be used for axis annotation relative to the current setting of cex. See par.</td>
</tr>
<tr>
<td>...</td>
<td>Arguments to be passed to plot.</td>
</tr>
</tbody>
</table>

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

```r
# 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, housekeeping 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**

```r
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 ploted. For “p95p05”, the trajectory of the ratio of 95th percentile to 5th percentile of the expression levels of all QC probes will be ploted.

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. `requireLog2=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 _positive_ 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 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

```r
# 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)
```

Description

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

```r
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

- **es**: ExpressionSet object of Sample probe profiles.
- **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.
- **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 _positive_ 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.
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>las</td>
<td>'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 <code>par</code>.</td>
</tr>
<tr>
<td>cex.axis</td>
<td>The magnification to be used for axis annotation relative to the current setting of <code>cex</code>. see <code>par</code>.</td>
</tr>
<tr>
<td>title</td>
<td>Figure title.</td>
</tr>
<tr>
<td>cex.legend</td>
<td>Font size of legend text.</td>
</tr>
<tr>
<td>cex.lab</td>
<td>The magnification to be used for x and y labels relative to the current setting of <code>cex</code>.</td>
</tr>
<tr>
<td>legendPosition</td>
<td>Position of legend. Possible values are “bottomright”, “bottom”, “bottomleft”, “left”, “topleft”, “top”, “topright”, “right” and “center”.</td>
</tr>
<tr>
<td>cut1</td>
<td>second horizontal line setting the cutoff for the ratio p95/p05. A ratio above this line indicates the corresponding array is good.</td>
</tr>
<tr>
<td>cut2</td>
<td>second horizontal line setting the cutoff for the ratio p95/p05. A ratio below this line indicates the corresponding array is bad.</td>
</tr>
<tr>
<td>sortFlag</td>
<td>logical. Indicates if arrays need to be sorted according to <code>Batch_Run_Date</code>, <code>ChipBarcode</code>, and <code>Chip_Address</code>.</td>
</tr>
<tr>
<td>varSort</td>
<td>A vector of phenotype variable names to be used to sort the samples of <code>es</code>.</td>
</tr>
<tr>
<td>timeFormat</td>
<td>A vector of time format for the possible time variables in <code>varSort</code>. The length of <code>timeFormat</code> should be the same as that of <code>varSort</code>. For non-time variable, the corresponding time format should be set to be equal to <code>NA</code>. The details of the time format for time variable can be found in the R function <code>strptime</code>.</td>
</tr>
<tr>
<td>verbose</td>
<td>logical. Determine if intermediate output need to be suppressed. By default <code>verbose=FALSE</code>, intermediate output will not be printed.</td>
</tr>
</tbody>
</table>

**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 row of the matrix 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**

```r
# generate simulated data set from conditional normal distribution
set.seed(1234567)
es.sim = genSimData.BayesNormal(nCpGs = 100,
nCases = 20, nControls = 20,
...Arguments to be passed to `plot`.```
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

dat  Expression data. Rows are gene probes; columns are arrays.

fileName  File name of output figure.

probs  quantiles (any real values between the interval [0,1]).

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. requireLog2=TRUE indicates probe expression levels will be log2 transformed. Otherwise, no transformation will be performed.

sortFlag  logical. sortFlag=TRUE indicates arrays will be sorted by the ascending order of MAD (median absolute deviation)

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 _positive_ number, defaulting to ’1’. see par.

main  Charater string. main title of the plot.

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.

...  Arguments to be passed to plot.

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 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)

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

es | ExpressionSet object of QC probe profiles.
hybName | character string. indicating the phenotype variable Hybridization_Name.
arrayType | A character string indicating if the correlations are calculated based on all arrays, arrays with replicates, or genetic control arrays.
GCid | A vector of character string. symbols for genetic control samples. The symbols can be more than one.
probs | A vector of probabilities specify the quantiles of correlations to be output.
col colors used for the image. see the function `heatmap.2` in R package gplots.

labelVariable A character string indicating how to label the arrays.

outFileName A character string. The name of output file.

title Title of the plot.

requireLog2 logical. `requireLog2=TRUE` indicates probe expression levels will be log2 transformed. Otherwise, no transformation will be performed.

plotOutPutFlag logical. `plotOutPutFlag=TRUE` indicates the plots will be output to pdf format files. Otherwise, the plots will not be output to external files.

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`.

keysize numeric value indicating the size of the key. see the function `heatmap.2` in R package gplots.

margins numeric vector of length 2 containing the margins. see the function `heatmap.2` in R package gplots.

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`.

... Arguments to be passed to `heatmap.2`.

Value

A list with 3 elements. 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 <stvc@channing.harvard.edu>, Benjamin Raby <rebar@channing.harvard.edu>

Examples

```r
# 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()

sortExpressionSet  Sort the order of samples for an ExpressionSet object

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.
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/xy", NA, NA)
	)

<table>
<thead>
<tr>
<th>Topic</th>
<th>Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>methods</td>
<td>genExprSet, 2</td>
</tr>
<tr>
<td></td>
<td>getPCAFunc, 4</td>
</tr>
<tr>
<td></td>
<td>glmWrapper, 6</td>
</tr>
<tr>
<td></td>
<td>lkmrWrapper, 9</td>
</tr>
<tr>
<td></td>
<td>lmFitPaired, 11</td>
</tr>
<tr>
<td></td>
<td>lmFitWrapper, 13</td>
</tr>
<tr>
<td></td>
<td>LumiBatch2Table, 16</td>
</tr>
<tr>
<td></td>
<td>pca2DPlot, 17</td>
</tr>
<tr>
<td></td>
<td>pca3DPlot, 19</td>
</tr>
<tr>
<td></td>
<td>plotCurves, 21</td>
</tr>
<tr>
<td></td>
<td>plotQCCurves, 23</td>
</tr>
<tr>
<td></td>
<td>plotSamplep95p05, 25</td>
</tr>
<tr>
<td></td>
<td>quantilePlot, 28</td>
</tr>
<tr>
<td></td>
<td>R2PlotFunc, 30</td>
</tr>
<tr>
<td></td>
<td>sortExpressionSet, 32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Topic</th>
<th>method</th>
</tr>
</thead>
<tbody>
<tr>
<td>method</td>
<td>genSimData.BayesNormal, 3</td>
</tr>
</tbody>
</table>

| methods    | genExprSet, 2                                                          |
|            | genSimData.BayesNormal, 3                                              |
|            | getPCAFunc, 4                                                           |
|            | glm, 6, 9                                                               |
|            | glmWrapper, 6                                                          |

| methods    | heatmap.2, 31                                                           |
|            | lkmrWrapper, 9                                                         |
|            | lmFitPaired, 11                                                         |
|            | lmFitWrapper, 13                                                       |
|            | LumiBatch2Table, 16                                                   |
|            | pca2DPlot, 17                                                          |
|            | pca3DPlot, 19                                                          |
|            | plot, 18, 20, 22, 24, 26, 27, 29, 31                                  |
|            | plotCurves, 21                                                         |
|            | plotQCCurves, 23                                                       |
|            | plotSamplep95p05, 25                                                   |
|            | quantilePlot, 28                                                       |
|            | R2PlotFunc, 30                                                         |
|            | sortExpressionSet, 32                                                  |

| methods    | strptime, 27, 31, 32                                                   |
|            | write.table, 16                                                        |
|            | write.table, 16                                                        |