Package ‘BatchQC’
March 22, 2017

Type Package
Title Batch Effects Quality Control Software
Version 1.2.1
Date 2016-11-23
Maintainer Solaiappan Manimaran <manimaran_1975@hotmail.com>
Description Sequencing and microarray samples often are collected or processed in multiple batches or at different times. This often produces technical biases that can lead to incorrect results in the downstream analysis. BatchQC is a software tool that streamlines batch preprocessing and evaluation by providing interactive diagnostics, visualizations, and statistical analyses to explore the extent to which batch variation impacts the data. BatchQC diagnostics help determine whether batch adjustment needs to be done, and how correction should be applied before proceeding with a downstream analysis. Moreover, BatchQC interactively applies multiple common batch effect approaches to the data, and the user can quickly see the benefits of each method. BatchQC is developed as a Shiny App. The output is organized into multiple tabs, and each tab features an important part of the batch effect analysis and visualization of the data. The BatchQC interface has the following analysis groups: Summary, Differential Expression, Median Correlations, Heatmaps, Circular Dendrogram, PCA Analysis, Shape, ComBat and SVA.

Author Solaiappan Manimaran <manimaran_1975@hotmail.com>, W. Evan Johnson <wej@bu.edu>, Heather Selby <selbyh@bu.edu>, Claire Ruberman <claireruberman@gmail.com>, Kwame Okrah <kwame.okrah@gmail.com>, Hector Corrada Bravo <hcorrada@gmail.com>

URL https://github.com/mani2012/BatchQC
BugReports https://github.com/mani2012/BatchQC/issues
License GPL (>= 2)
Depends R (>= 3.3.0)
Suggests  testthat

Imports  utils, rmarkdown, knitr, pander, gplots, MCMCpack, shiny, sva,
corpcor, moments, matrixStats, ggvis, d3heatmap, reshape2,
limma, grDevices, graphics, stats, methods

biocViews  BatchEffect, GraphAndNetwork, Microarray,
PrincipalComponent, Sequencing, Software, Visualization,
QualityControl, RNASEq, Preprocessing, DifferentialExpression

SystemRequirements  pandoc (http://pandoc.org/installing.html) for
            generating reports from markdown files.

VignetteBuilder  knitr

RoxygenNote  5.0.1

NeedsCompilation  no

R topics documented:

batchQC                     3
BatchQCout-class            4
batchQC_analyze             4
batchqc_circosplot          5
batchQC_condition_adjusted  6
batchqc_correlation         6
batchqcCorscatter           7
batchqcExplainedVariation   8
batchQCFilter_genes         8
batchQC_fsva_adjusted       9
batchqc_heatmap             10
batchQC_num.sv              11
batchqc_pca                 11
batchqc_pca_svd             12
batchqc_pc_explained_variation 13
batchQC_shapeVariation      14
batchQC_sva                 14
batchQC_svregrress_adjusted 15
batchtest                   16
combatPlot                  16
example_batchqc_data        17
getShinyInput               18
getShinyInputCombat         19
getShinyInputOrig           19
getShinyInputSVA            20
getShinyInputSVAf           20
getShinyInputSVAR           21
gnormalize                 21
log2CPM                    22
makeSVD                    22
pcRes                      23
plotPC                     23
protein_example_data       24
rnaseq_sim                  24
setShinyInput              25
batchQC

Checks for presence of batch effect and creates a html report with information including whether the batch needs to be adjusted

Description

Checks for presence of batch effect and creates a html report with information including whether the batch needs to be adjusted

Usage

```
batchQC(dat, batch, condition = NULL, report_file = "batchqc_report.html", report_dir = ".", report_option_binary = "111111111", view_report = FALSE, interactive = TRUE, batchqc_output = FALSE, log2cpm_transform = FALSE)
```

Arguments

- `dat` Given data or simulated data from rnaseq_sim()
- `batch` Batch covariate
- `condition` Covariates or conditions of interest besides batch
- `report_file` Output report file name
- `report_dir` Output report directory path
- `report_option_binary` 9 bits Binary String representing the plots to display and hide in the report
- `view_report` when TRUE, opens the report in a browser
- `interactive` when TRUE, opens the interactive shinyApp
- `batchqc_output` when TRUE, creates BatchQCout object in batchqc_output.rda R object file
- `log2cpm_transform` when TRUE, transforms the data using log2CPM - log2 Counts Per Million transformation function

Value

- `outputfile` Report file generated by batchQC
Examples

```r
nbatch <- 3
cmpd <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=npercond,
                     basemean=10000, ggstep=50, bstep=2000, ccstep=800,
                     basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
batchQC(data.matrix, batch=batch, condition=condition, view_report=FALSE,
        interactive=FALSE)
```

BatchQCout-class

The BatchQC output class to output BatchQC results

Description

Contains all currently-supported BatchQC output data classes:

Details

slots:

- `batchqc_ev`: a single object of class list
- `pca`: a single object of S3 class prcomp

batchQC_analyze

Checks for presence of batch effect and reports whether the batch needs to be adjusted

Description

Checks for presence of batch effect and reports whether the batch needs to be adjusted

Usage

```r
batchQC_analyze(data.matrix, batch, mod = NULL)
```

Arguments

- `data.matrix`: Given data or simulated data from rnaseq_sim()
- `batch`: Batch covariate
- `mod`: Model matrix for outcome of interest and other covariates besides batch

Value

- `pca`: Principal Components Analysis object of the data
Examples

nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=npercond,
                         basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
                         basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
pdata <- data.frame(batch, condition)
modmatrix = model.matrix(~as.factor(condition), data=pdata)
batchQC_analyze(data.matrix, batch, mod=modmatrix)

batchqc_circosplot  Produce Circos plot

Description

Produce Circos plot

Usage

batchqc_circosplot(dat, batch, AggMethod)

Arguments

dat  Given data or simulated data from rnaseq_sim()
batch  Batch covariate
AggMethod  Aggregation Method

Value

Generates Circular Dendrogram plot for the given data

Examples

nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=npercond,
                         basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
                         basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
batchqc_circosplot(data.matrix, batch=batch, AggMethod='complete')
**batchQC_condition_adjusted**

*Returns adjusted data after remove the variation across conditions*

**Description**

Returns adjusted data after remove the variation across conditions

**Usage**

```
batchQC_condition_adjusted(data.matrix, batch, condition)
```

**Arguments**

- `data.matrix`: Given data or simulated data from rnaseq_sim()
- `batch`: Batch covariate
- `condition`: Condition covariate of interest

**Value**

Adjusted data after remove the variation across conditions

**Examples**

```
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=npercond,
                         basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
batchQC_condition_adjusted(data.matrix, batch, condition)
```

---

**batchqc_correlation**  
*Produce correlation heatmap plot*

**Description**

Produce correlation heatmap plot

**Usage**

```
batchqc_correlation(data.matrix, batch, mod = NULL)
```

**Arguments**

- `data.matrix`: Given data or simulated data from rnaseq_sim()
- `batch`: Batch covariate
- `mod`: Model matrix for outcome of interest and other covariates besides batch
**Value**

Correlation heatmap plot

**Examples**

```r
nbchat <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbchat=nbatch, ncond=ncond, npercond=
     npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
     basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
bchat <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
pdata <- data.frame(bchat, condition)
modmatrix = model.matrix(~as.factor(condition), data=pdata)
nbatchqc_correlation(data.matrix, batch, mod=modmatrix)
```

**Description**

Produce Median Correlation plot

**Usage**

`batchqc_corscatter(data.matrix, batch, mod = NULL)`

**Arguments**

- `data.matrix`: Given data or simulated data from `rnaseq_sim()`
- `batch`: Batch covariate
- `mod`: Model matrix for outcome of interest and other covariates besides batch

**Value**

Median Correlation plot

**Examples**

```r
nbchat <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbchat=nbatch, ncond=ncond, npercond=
     npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
     basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
bchat <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
pdata <- data.frame(bchat, condition)
modmatrix = model.matrix(~as.factor(condition), data=pdata)
nbatchqc_corscatter(data.matrix, batch, mod=modmatrix)
```
batchqc_explained_variation

*Returns a list of explained variation by batch and condition combinations*

**Description**

Returns a list of explained variation by batch and condition combinations

**Usage**

`batchqc_explained_variation(data.matrix, condition, batch)`

**Arguments**

- `data.matrix` Given data or simulated data from `rnaseq_sim()`
- `condition` Condition covariate of interest
- `batch` Batch covariate

**Value**

List of explained variation by batch and condition

**Examples**

```r
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=
    npercond, basemean=10000, gstep=50, bbstep=2000, ccstep=800,
basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
batchQC_filter_genes(data.matrix, batch)
```

batchQC_filter_genes

*Returns a dataset after filtering genes of zero variance across batch and condition combinations*

**Description**

Returns a dataset after filtering genes of zero variance across batch and condition combinations

**Usage**

`batchQC_filter_genes(data.matrix, batch, condition)`
Arguments

- **data.matrix**: Given data or simulated data from `rnaseq_sim()`
- **batch**: Batch covariate
- **condition**: Condition covariate of interest

Value

Filtered dataset after filtering genes of zero variance across batch and condition combinations

Examples

```r
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenoes=50, nbatch=nbatch, ncond=ncond, npercond=npercond,
                         basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
filtered.data <- batchQC_filter_genes(data.matrix, batch, condition)
```
batchqc_heatmap

Produce heatmap plots for the given data

Description

Produce heatmap plots for the given data

Usage

batchqc_heatmap(data.matrix, batch, mod = NULL, max_display = 50)

Arguments

data.matrix Given data or simulated data from rnaseq_sim()
batch Batch covariate
mod Model matrix for outcome of interest and other covariates besides batch
max_display Maximum number of rows to display in heat map

Value

Heatmap plots for the given data

Examples

nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=
npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
data <- data.frame(batch, condition)
modmatrix = model.matrix(~as.factor(condition), data=data)
sva.object <- batchQC_sva(data.matrix, mod=modmatrix)
batchQC_fsva_adjusted(data.matrix, modmatrix, sva.object)

nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=
npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
data <- data.frame(batch, condition)
modmatrix = model.matrix(~as.factor(condition), data=data)
batchqc_heatmap(data.matrix, batch, mod=modmatrix)
batchQC_num.sv

Returns the number of surrogate variables to estimate in the model using a permutation based procedure

Description
Returns the number of surrogate variables to estimate in the model using a permutation based procedure

Usage
batchQC_num.sv(data.matrix, modmatrix)

Arguments

- **data.matrix**: Given data or simulated data from rnaseq_sim()
- **modmatrix**: Model matrix for outcome of interest and other covariates besides batch

Value
Number of Surrogate variables found

Examples

```r
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=npercond,
                         npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
                         basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
pdata <- data.frame(batch, condition)
modmatrix = model.matrix(~as.factor(condition), data=pdata)
batchQC_num.sv(data.matrix, modmatrix)
```

batchqc_pca

Performs principal component analysis and produces plot of the first two principal components

Description
Performs principal component analysis and produces plot of the first two principal components

Usage
batchqc_pca(data.matrix, batch, mod = NULL)
Arguments

- **data.matrix**
  
  Given data or simulated data from `rnaseq_sim()`

- **batch**

  Batch covariate

- **mod**

  Model matrix for outcome of interest and other covariates besides batch

Value

PCA object from principal component analysis

Examples

```r
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngen=50, nbatch=nbatch, ncond=ncond, npercond=npercond, basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
pdata <- data.frame(batch, condition)
modmatrix = model.matrix(~as.factor(condition), data=pdata)
batchqc_pca(data.matrix, batch, mod=modmatrix)
```

Description

Performs PCA svd variance decomposition and produces plot of the first two principal components

Usage

```r
batchqc_pca_svd(data.matrix, batch, mod = NULL)
```

Arguments

- **data.matrix**

  Given data or simulated data from `rnaseq_sim()`

- **batch**

  Batch covariate

- **mod**

  Model matrix for outcome of interest and other covariates besides batch

Value

res PCA list with two components v and d.
Examples

```r
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=npercond,
                         npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
                         basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
pdata <- data.frame(batch, condition)
modmatrix = model.matrix(~as.factor(condition), data=pdata)
batchqc_pca_svd(data.matrix, batch, mod=modmatrix)
```

returns explained variation for each principal components

Description

returns explained variation for each principal components

Usage

`batchqc_pc_explained_variation(pcs, vars, condition, batch)`

Arguments

- `pcs`: Principal components in the given data
- `vars`: Variance of the Principal components in the given data
- `condition`: Condition covariate of interest
- `batch`: Batch covariate

Value

Explained variation table for each principal components

Examples

```r
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=npercond,
                         npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
                         basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
pdata <- data.frame(batch, condition)
modmatrix = model.matrix(~as.factor(condition), data=pdata)
pca <- batchqc_pca(data.matrix, batch, mod=modmatrix)
pcs <- t(data.frame(pca$x))
batchqc_pc_explained_variation(pcs, pca$sdev^2, condition, batch)
```
Perform Mean and Variance batch variation analysis

batchQC_shapeVariation

Description
Perform Mean and Variance batch variation analysis

Usage
batchQC_shapeVariation(data, groups, plot = FALSE, groupCol = NULL)

Arguments
- data: Given data
- groups: a character vector indicating sample group membership
- plot: Indicate whether to generate plot
- groupCol: group color

Value
Mean and Variance batch variation Overall and Pairwise p-values

Examples
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=
npercond, base.mean=10000, ggstep=50, bbstep=2000, ccstep=800,
basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
batchQC_shapeVariation(data.matrix, groups=batch)

batchQC_sva

Estimate the surrogate variables using the 2 step approach proposed by Leek and Storey 2007

Description
Estimate the surrogate variables using the 2 step approach proposed by Leek and Storey 2007

Usage
batchQC_sva(data.matrix, modmatrix)

Arguments
- data.matrix: Given data or simulated data from rnaseq_sim()
- modmatrix: Model matrix for outcome of interest and other covariates besides batch
Regress the surrogate variables out of the expression data

Usage

`batchQC_svregress_adjusted(data.matrix, modmatrix, sva.object)`

Arguments

data.matrix: Given data or simulated data from `rnaseq_sim()`
modmatrix: Model matrix for outcome of interest and other covariates besides batch
sva.object: SVA object

Value

Surrogate variables regress adjusted data

Examples

```r
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngen=50, nbatch=nbatch, ncond=ncond, npercond=npercond,
basedisp=100, bdispstep=-10, swvar=100, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
pdata <- data.frame(batch, condition)
modmatrix = model.matrix(~as.factor(condition), data=pdata)
sva.object <- batchQC_sva(data.matrix, modmatrix)
batchQC_svregress_adjusted(data.matrix, modmatrix, sva.object)
```
**batchtest**

*Performs test to check whether batch needs to be adjusted*

**Description**

Performs test to check whether batch needs to be adjusted

**Usage**

```r
batchtest(pca, batch, mod = NULL)
```

**Arguments**

- `pca`: PCA object from principal component analysis
- `batch`: Batch covariate
- `mod`: Model matrix for outcome of interest and other covariates besides batch

**Value**

Summary of linear regression of first five principal components

**Examples**

```r
nbatch <- 3
cond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=cond, npercond=npercond,
                         npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
                         basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=cond*npercond)
condition <- rep(rep(1:cond, each=npercond), nbatch)
pdata <- data.frame(batch, condition)
modmatrix = model.matrix(~as.factor(condition), data=pdata)
pca <- batchqc_pca(data.matrix, batch, mod=modmatrix)
batchtest(pca, batch, mod=modmatrix)
```

---

**combatPlot**

*Adjust for batch effects using an empirical Bayes framework ComBat allows users to adjust for batch effects in datasets where the batch covariate is known, using methodology described in Johnson et al. 2007. It uses either parametric or non-parametric empirical Bayes frameworks for adjusting data for batch effects. Users are returned an expression matrix that has been corrected for batch effects. The input data are assumed to be cleaned and normalized before batch effect removal.*
Description

Adjust for batch effects using an empirical Bayes framework ComBat allows users to adjust for batch effects in datasets where the batch covariate is known, using methodology described in Johnson et al. 2007. It uses either parametric or non-parametric empirical Bayes frameworks for adjusting data for batch effects. Users are returned an expression matrix that has been corrected for batch effects. The input data are assumed to be cleaned and normalized before batch effect removal.

Usage

`combatPlot(dat, batch, mod = NULL, par.prior = TRUE, prior.plots = TRUE)`

Arguments

dat Genomic measure matrix (dimensions probe x sample) - for example, expression matrix
batch Batch covariate (only one batch allowed)
mod Model matrix for outcome of interest and other covariates besides batch
par.prior (Optional) TRUE indicates parametric adjustments will be used, FALSE indicates non-parametric adjustments will be used
prior.plots (Optional) TRUE give prior plots with black as a kernel estimate of the empirical batch effect density and red as the parametric

Value
data A probe x sample genomic measure matrix, adjusted for batch effects.

Examples

```r
nbatch <- 3
cond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=
npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800,
basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
batch <- rep(1:nbatch, each=ncond*npercond)
condition <- rep(rep(1:ncond, each=npercond), nbatch)
pdata <- data.frame(batch, condition)
mod = model.matrix(~as.factor(condition), data = pdata)
combatPlot(data.matrix, batch, mod=mod)
```

example_batchqc_data Batch and Condition indicator for signature data captured when activating different growth pathway genes in human mammary epithelial cells.

Description

This data consists of three batches and ten different conditions corresponding to control and nine different pathways.

This data consists of three batches and ten different conditions corresponding to control and nine different pathways.
**Usage**

batch_indicator

signature_data

**Format**

A data frame with 89 rows and 2 variables:

**V1** Batch Indicator

**V2** Condition (Pathway) Indicator

**Value**

Batch indicator object

Signature data

**Source**

GEO accession: GSE73628

GEO accession: GSE73628

getShinyInput Getter function to get the shinyInput option

**Description**

Getter function to get the shinyInput option

**Usage**

getShinyInput()

**Value**

shinyInput option

**Examples**

getShinyInput()
getShinyInputCombat  
*Getter function to get the shinyInputCombat option*

**Description**

Getter function to get the shinyInputCombat option

**Usage**

getShinyInputCombat()

**Value**

shinyInputCombat option

**Examples**

getShinyInputCombat()

---

getShinyInputOrig  
*Getter function to get the shinyInputOrig option*

**Description**

Getter function to get the shinyInputOrig option

**Usage**

getShinyInputOrig()

**Value**

shinyInputOrig option

**Examples**

getShinyInputOrig()
getShinyInputSVA

**Description**

Getter function to get the shinyInputSVA option

**Usage**

getShinyInputSVA()

**Value**

shinyInputSVA option

**Examples**

getShinyInputSVA()

getShinyInputSVAf

**Description**

Getter function to get the shinyInputSVAf option

**Usage**

getShinyInputSVAf()

**Value**

shinyInputSVAf option

**Examples**

getShinyInputSVAf()
getShinyInputSVAr

**Description**

Getter function to get the shinyInputSVAr option

**Usage**

getShinyInputSVAr()

**Value**

shinyInputSVAr option

**Examples**

getShinyInputSVAr()


gnormalize

**Description**

Perform Genewise Normalization of the given data matrix

**Usage**

gnormalize(dat)

**Arguments**

dat Given data matrix

**Value**

gnormdata Genewise Normalized data matrix

**Examples**

dat <- matrix(1:10, 2)
gnormdata <- gnormalize(dat)
log2CPM

Compute log2(counts per mil reads) and library size for each sample

Description
Compute log2(counts per mil reads) and library size for each sample

Usage
log2CPM(qcounts, lib.size = NULL)

Arguments
- qcounts: quantile normalized counts
- lib.size: default is colsums(qcounts)

Value
List containing log2(quantile counts per mil reads) and library sizes

Examples
nbatch <- 3
ncond <- 2
npercond <- 10
data.matrix <- rnaseq_sim(ngenes=50, nbatch=nbatch, ncond=ncond, npercond=npercond, basemean=10000, ggstep=50, bbstep=2000, ccstep=800, basedisp=100, bdispstep=-10, swvar=1000, seed=1234)
data.matrix <- as.matrix(data.matrix)
log2CPM(data.matrix)

makeSVD

Compute singular value decomposition

Description
Compute singular value decomposition

Usage
makeSVD(x)

Arguments
- x: matrix of genes by sample (ie. the usual data matrix)

Value
Returns a list of svd components v and d
pcRes

Compute variance of each principal component and how they correlate with batch and cond

Description
Compute variance of each principal component and how they correlate with batch and cond

Usage
pcRes(v, d, condition = NULL, batch = NULL)

Arguments
v from makeSVD
d from makeSVD
condition factor describing experiment
batch factor describing batch

Value
A dataframe containing variance, cum. variance, cond.R-sqrd, batch.R-sqrd

plotPC
Plot first 2 principal components

Description
Plot first 2 principal components

Usage
plotPC(v, d, ...)

Arguments
v from makeSVD
d from makeSVD
... pass options to internal plot fct.

Value
a plot
protein_example_data  Batch and Condition indicator for protein expression data

Description
This data consists of two batches and two conditions corresponding to case and control for the protein expression data

Usage
protein_sample_info
protein_data

Format
A data frame with 24 rows and 4 variables:

- **Arrayname**  Array Name
- **samplename**  Sample Name
- **Batch**  Batch Indicator
- **category**  Condition (Case vs Control) Indicator

Value
Protein data sample info
Protein data

rnaseq_sim  Generate simulated count data with batch effects for ngenes

Description
Generate simulated count data with batch effects for ngenes

Usage
rnaseq_sim(ngenes = 50, nbatch = 3, ncond = 2, npercond = 10,
basemean = 10000, ggstep = 50, bbstep = 2000, ccstep = 800,
basedisp = 100, bdispstep = 10, swvar = 1000, seed = 1000)
Arguments

- `ngen`: Number of genes to simulate
- `nbatch`: Number of batches to simulate
- `ncond`: Number of conditions to simulate
- `npercond`: Number of samples per condition per batch to simulate
- `basemean`: Base mean
- `ggstep`: Gene to Gene step variation
- `bbstep`: Batch to Batch step variation
- `ccstep`: Condition to Condition step variation
- `basedisp`: Base Dispersion
- `bdispstep`: Batch to Batch Dispersion step variation
- `swvar`: Sample-wise extra variation
- `seed`: Random seed for reproducibility

Value

RNA Seq count data matrix

Examples

```r
rnaseq_sim()
rnaseq_sim(ngenes=100, nbatch=5, seed=1234)
rnaseq_sim(ngenes=100, nbatch=3, ncond=2, npercond=10, basemean=10000,
  ggstep=50, bbstep=20000, ccstep=8000, basedisp=100, bdispstep=10,
  swvar=1000, seed=1234)
```

---

**setShinyInput**

*Setter function to set the shinyInput option*

Description

Setter function to set the shinyInput option

Usage

```r
setShinyInput(x)
```

Arguments

- `x`: shinyInput option

Value

shinyInput option

Examples

```r
setShinyInput(NULL)
```
setShinyInputCombat  Setter function to set the shinyInputCombat option

Description
Setter function to set the shinyInputCombat option

Usage
setShinyInputCombat(x)

Arguments
x          shinyInputCombat option

Value
shinyInputCombat option

Examples
setShinyInputCombat(NULL)

setShinyInputOrig  Setter function to set the shinyInputOrig option

Description
Setter function to set the shinyInputOrig option

Usage
setShinyInputOrig(x)

Arguments
x          shinyInputOrig option

Value
shinyInputOrig option

Examples
setShinyInputOrig(NULL)
setShinyInputSVA  

Setter function to set the shinyInputSVA option

Description
Setter function to set the shinyInputSVA option

Usage
setShinyInputSVA(x)

Arguments
x  shinyInputSVA option

Value
shinyInputSVA option

Examples
setShinyInputSVA(NULL)

---

setShinyInputSVAf  

Setter function to set the shinyInputSVAf option

Description
Setter function to set the shinyInputSVAf option

Usage
setShinyInputSVAf(x)

Arguments
x  shinyInputSVAf option

Value
shinyInputSVAf option

Examples
setShinyInputSVAf(NULL)
Description

Setter function to set the shinyInputSVAr option

Usage

setShinyInputSVAr(x)

Arguments

x  shinyInputSVAr option

Value

shinyInputSVAr option

Examples

setShinyInputSVAr(NULL)
Index

*Topic datasets*
  example_batchqc_data, 17
  protein_example_data, 24

batch_indicator (example_batchqc_data), 17
batchQC, 3
batchQC_analyze, 4
batchqc_circosplot, 5
batchQC_condition_adjusted, 6
batchqc_correlation, 6
batchqcCorscatter, 7
batchqc_explained_variation, 8
batchQC_filter_genes, 8
batchQC_fsve_adjusted, 9
batchqc_heatmap, 10
batchQC_num.sv, 11
batchqc_pc_explained_variation, 13
batchqc_pca, 11
batchqc_pca_svd, 12
batchQC_shapeVariation, 14
batchQC_sva, 14
batchQC_svregress_adjusted, 15
BatchQCout-class, 4
batchtest, 16

combatPlot, 16
example_batchqc_data, 17

getShinyInput, 18
getShinyInputCombat, 19
getShinyInputOrig, 19
getShinyInputSVA, 20
getShinyInputSVAf, 20
getShinyInputSVAR, 21
gnormalize, 21

log2CPM, 22

makeSVD, 22

pcRes, 23
plotPC, 23
protein_data (protein_example_data), 24

protein_example_data, 24
protein_sample_info (protein_example_data), 24
rnaseq_sim, 24

setShinyInput, 25
setShinyInputCombat, 26
setShinyInputOrig, 26
setShinyInputSVA, 27
setShinyInputSVAf, 27
setShinyInputSVAR, 28
signature_data (example_batchqc_data), 17