Package 'inSilicoMerging'

October 8, 2014

Version 1.8.7

Date 2014-09-09

Title Collection of Merging Techniques for Gene Expression Data

Description Collection of techniques to remove inter-
study bias when combining gene expression data originating from different studies.

Author Jonatan Taminau <jtaminau@gmail.com>

Maintainer Quentin De Clerck <qdeclerc@vub.ac.be>, David Steenhoff <davidsteenhoff@insilicodb.org>

Depends R (>= 2.11.1), Biobase, DWD

Suggests BiocGenerics, inSilicoDb

Collate util.R dwd.R xpn.R merge.R mergeBMC.R mergeGENENORM.R
mergeCOMBAT.R mergeNONE.R mergeXPN.R mergeDWD.R color.R
plotMDS.R plotRLE.R plotGeneWiseBoxPlot.R test_inSilicoMerging_package.R

biocViews Microarray

License GPL-2

URL http://insilicodb.com/

R topics documented:

`inSilicoMerging-package` *Collection of Merging Techniques for Gene Expression Data.*

Description

This package provides a collection of techniques to remove inter-study bias when combining gene expression data originating from different studies.

See Also

`merge` `plotMDS` `plotRLE` `plotGeneWiseBoxPlot`

`merge` *General method to merge different ExpressionSets*

Description

General method to merge different ExpressionSets by applying different techniques to remove inter-study bias.

Usage

```
merge(esets, method=NA);
```

Arguments

`esets`	List of ExpressionSet objects.
`method`	Merging method aimed at removing inter-study bias. Possible options are: BMC, COMBAT, DWD, GENENORM and XPN. If none are specified, the merging More information about each method is given below in the details.

Details

Currently the following different merging techniques are provided:

'BMC': In [1] they successfully applied a technique similar to z-score normalization for merging breast cancer datasets. They transformed the data by batch mean-centering, which means that the mean is subtracted.

'COMBAT': Empirical Bayes [2] (also called EJLR or COMBAT) is a method that estimates the parameters of a model for mean and variance for each gene and then adjusts the genes in each batch to meet the assumed model. The parameters are estimated by pooling information from multiple genes in each batch.

'DWD': In [3] they propose to use Distance Weighted Discrimination to find the hyperplane that separates the expression values of two studies. Assuming that the variation due to studies originating from different labs is bigger than any biological variation present in the data a translation in the direction of the normal vector of the hyperplane is used to remove bias.

'GENENORM': One of the simplest mathematical transformations to make datasets more comparable is z-score normalization. In this method, for each gene expression value in each study separately all values are altered by subtracting the mean of the gene in that dataset divided by its standard deviation.

NA: Combine esets without any additional transformation. Similar to 'combine' function.

'XPN': The basic idea behind the cross-platform normalization [4] approach is to find blocks (clusters) of genes and samples in both studies that have similar expression characteristics. In XPN, a gene measurement can be considered as a scaled and shifted block mean.

Note that after using any of those methods the resulting merged dataset only contains the common list of genes/probes between all studies.

Value

A (merged) ExpressionSet object.

References

[1] A. Sims, *et al.*, The removal of multiplicative, systematic bias allows integration of breast cancer gene expression datasets - improving meta-analysis and prediction of prognosis, *BMC Medical Genomics*, vol. 1, no. 1, p. 42, 2008.

[2] C. Li and A. Rabinovic, Adjusting batch effects in microarray expression data using empirical bayes methods, *Biostatistics*, vol. 8, no. 1, pp. 118-127, 2007.

[3] M. Benito, *et al.*, Adjustment of systematic microarray data biases, *Bioinformatics*, vol. 20, no. 1, pp. 105-114, 2004.

[4] A. A. Shabalin, *et al.*, Merging two gene-expression studies via cross-platform normalization, *Bioinformatics*, vol. 24, no. 9, pp. 1154-1160, 2008.

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE18842", "GPL570", norm="FRMA", features="GENE");
eset2 = getDataset("GSE31547", "GPL96",  norm="FRMA", features="GENE");
esets = list(eset1,eset2);

# merge them using different methods:
library(inSilicoMerging);
eset_FRMA = merge(esets);
eset_COMBAT = merge(esets, method="COMBAT");
```

plotGeneWiseBoxPlot *Create gene-wise boxplot from (merged) ExpressionSet*

Description

Gene-wise boxplots describe the gene-wise distribution of samples. Sample can be grouped together using the batchLabel parameter and can be coloroed usig the colLabel parameter for optimal visualization of the possible batch effects.

Usage

```
plotGeneWiseBoxPlot(eset, colLabel, batchLabel, gene=NULL, legend=TRUE, file=NULL, ...)
```

Arguments

eset	ExpressionSet object.
colLabel	colname in pData(eset) to retrieve information for the labeling of samples with a color. All samples with the same value in pData(eset)[,colLabel] will share the same color.
batchLabel	colname in pData(eset) to retrieve information for the grouping of samples. All samples with the same value in pData(eset)[,batchLabel] and with the same color will be grouped together.
gene	Gene for which the boxplot will be created. If not specified a random gene will be selected.
legend	If TRUE a legend will be provided next to the gene-wise box plot.
file	If defined, the resulting plot will be stored as a pdf file instead of shown interactively.
...	Additional parameters for the 'plot' function (e.g. 'main').

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE18842", "GPL570", norm="FRMA", features="gene");
eset2 = getDataset("GSE31547", "GPL96",  norm="FRMA",  features="gene");
esets = list(eset1,eset2);

# merge them using no additional merging technique and the COMBAT method:
library(inSilicoMerging)
eset_FRMA = merge(esets);
eset_COMBAT = merge(esets, method="COMBAT");

# check available annotations:
colnames(pData(eset_FRMA))
table(pData(eset_FRMA)[,"Disease"]);
table(pData(eset_FRMA)[,"Study"]);
```

```
# Visual inspection of a random gene in the two merged datasets
gene = sample(rownames(exprs(eset_FRMA)), 1)
plotGeneWiseBoxPlot(eset_FRMA, colLabel="Disease", batchLabel="Study", gene=gene)
plotGeneWiseBoxPlot(eset_COMBAT, colLabel="Disease", batchLabel="Study", gene=gene)
```

---

plotMDS *Create double-labeled MDS plot from (merged) ExpressionSet*

Description

Create Multidimensional Scaling (MDS) plot from ExpressionSet. Very similar to Principal Component Analysis (PCA) plots all samples are plotted in a two-dimensional space where both axis represent the two principle axis of expression variation. In this plot each sample can be labeled with a color and with a symbol.

Usage

```
plotMDS(eset, colLabel, symLabel, legend=TRUE, file=NULL, ...)
```

Arguments

eset	ExpressionSet object.
colLabel	colname in pData(eset) to retrieve information for the labeling of samples with a color. All samples with the same value in pData(eset)[,colLabel] will share the same color.
symLabel	colname in pData(eset) to retrieve information for the labeling of samples with a symbol. All samples with the same value in pData(eset)[,symLabel] will share the same symbol.
legend	If TRUE a legend will be provided next to the MDS plot for both colLabel and symlabel.
file	If defined, the resulting plot will be stored as a pdf file instead of shown interactively.
...	Additional parameters for the 'plot' function (e.g. 'main').

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE18842", "GPL570", norm="FRMA", features="gene");
eset2 = getDataset("GSE31547", "GPL96",  norm="FRMA", features="gene");
esets = list(eset1,eset2);

# merge them using no additional merging technique and the COMBAT method:
library(inSilicoMerging)
```

```
eset_FRMA = merge(esets);
eset_COMBAT = merge(esets, method="COMBAT");

# check available annotations:
colnames(pData(eset_FRMA))
table(pData(eset_FRMA)[,"Disease"]);
table(pData(eset_FRMA)[,"Study"]);

# Visual inspection of the two merged datasets through an MDS plot
plotMDS(eset_FRMA, colLabel="Disease", symLabel="Study")
plotMDS(eset_COMBAT, colLabel="Disease", symLabel="Study")
```

`plotRLE` *Create RLE plot from (merged) ExpressionSet*

Description

Create relative log expression (RLE) plot from ExpressionSet. RLE plots were initially proposed to measure the overall quality of a dataset but can also be used to visualize the presence of unwanted batch effects in the data.

Usage

```
plotRLE(eset, colLabel, legend=TRUE, file=NULL, ...)
```

Arguments

`eset`	ExpressionSet object.
`colLabel`	colname in pData(eset) to retrieve information for the labeling of samples with a color. All samples with the same value in pData(eset)[,colLabel] will share the same color.
`legend`	If TRUE a legend will be provided next to the RLE plot.
`file`	If defined, the resulting plot will be stored as a pdf file instead of shown interactively.
`...`	Additional parameters for the 'plot' function (e.g. 'main').

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE18842", "GPL570", norm="FRMA", features="gene");
eset2 = getDataset("GSE31547", "GPL96",  norm="FRMA", features="gene");
esets = list(eset1,eset2);

# merge them using no additional merging technique and the COMBAT method:
library(inSilicoMerging)
```

```
eset_FRMA = merge(esets);
eset_COMBAT = merge(esets, method="COMBAT");

# check available annotations:
colnames(pData(eset_FRMA))
table(pData(eset_FRMA)[,"Disease"]);
table(pData(eset_FRMA)[,"Study"]);

# Visual inspection of the two merged datasets through an MDS plot
plotRLE(eset_FRMA, colLabel="Disease")
plotRLE(eset_COMBAT, colLabel="Disease")
```

Index