Package 'inSilicoMerging'

September 24, 2012

Version 1.0.8

Date 2012-03-15

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@vub.ac.be>, Stijn Meganck <smeganck@vub.ac.be>, Cosmin Lazar <vlazar@vub.ac.be>

Maintainer Jonatan Taminau <jtaminau@vub.ac.be>, Stijn Meganck <smeganck@vub.ac.be>, Cosmin Lazar <vlazar@vub.ac.be>

Depends R (>= 2.11.1), Biobase, DWD, limma

Imports amap, moments

Suggests BiocGenerics, inSilicoDb

Collate zzz.R util.R dwd.R xpn.R merge.R mergeBMC.R mergeGENENORM.R mergeGENESHIFT.R mergeCOMBAT.R mergeNONE.R mergeXPN.R mergeDWD.R color.R plotMDS.R plotRLE.R plotDendrogram.R plotGeneWiseBoxPlot.R plotGeneWiseDensity.R measureAsymmetry.R measureSamplesOverlap.R measureGenesOverlap.R measureSignificantGenesOverlap.R measureSamplesMeanCorrCoef.R measureGenesMeanCorrCoef.R test_inSilicoMerging_package.R

biocViews Microarray

License GPL-2

URL http://insilico.ulb.ac.be/insilico-project/

R topics documented:

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

Description

This package provides a collection of techniques to remove batch effects when combining gene expression data originating from different studies.

See Also

`merge`

`measureAsymmetry` *Calculate asymmetry index for (merged) ExpressionSet*

Description

measureAsymmetry compares the distribution of samples asymmetry, quantified by the skewness, before and after batch effect removal. This index should have a value close to 0

Usage

```
measureAsymmetry(eset, batchAnnot="Batch")
```

Arguments

`eset`	ExpressionSet object.
`batchAnnot`	Column in pData(eset) containing the batch information.

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
```

```
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Calculate asymmetry index for both datasets
measureAsymmetry(eset_NONE, batchAnnot="Study");
measureAsymmetry(eset_COMBAT, batchAnnot="Study");
```

`measureGenesMeanCorrCoef`

Calculate gene mean correlation coefficient between two independent studies before and after applying batch effect removal methods.

Description

Calculate gene mean correlation coefficient between two independent studies before and after applying batch effect removal methods. Methods that least affects the data should be preferred.

Usage

```
measureGenesMeanCorrCoef(esetBEFORE, esetAFTER, batchAnnot="Batch")
```

Arguments

`esetBEFORE`	ExpressionSet of combined study before applying batch removal method.
`esetAFTER`	ExpressionSet of combined study after applying batch removal method.
`batchAnnot`	Column in pData(eset) containing the batch information.

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Calculate gene mean correlation coefficient for both datasets
measureGenesMeanCorrCoef(eset_NONE, eset_COMBAT, batchAnnot="Study");
```

`measureGenesOverlap` *Calculate Genes Overlap index for (merged) ExpressionSet*

Description

Measures the expected overlap of genes. The higher this overlap, the better the integration process.

Usage

```
measureGenesOverlap(eset, batchAnnot="Batch")
```

Arguments

`eset`	ExpressionSet object.
`batchAnnot`	Column in pData(eset) containing the batch information.

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Calculate Genes Overlap index for both datasets
measureGenesOverlap(eset_NONE, batchAnnot="Study");
measureGenesOverlap(eset_COMBAT, batchAnnot="Study");
```

`measureSamplesMeanCorrCoef`
Calculate sample mean correlation coefficient between two independent studies before and after applying batch effect removal methods.

Description

Calculate sample mean correlation coefficient between two independent studies before and after applying batch effect removal methods. Methods that least affects the data should be preferred.

Usage

```
measureSamplesMeanCorrCoef(esetBEFORE, esetAFTER, batchAnnot="Batch")
```

Arguments

esetBEFORE	ExpressionSet of combined study before applying batch effect removal method.
esetAFTER	ExpressionSet of combined study after applying batch effect removal method.
batchAnnot	Column in pData(eset) containing the batch information.

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Calculate sample mean correlation coefficient for both datasets
measureSamplesMeanCorrCoef(eset_NONE, eset_COMBAT, batchAnnot="Study");
```

---

measureSamplesOverlap *Calculate Samples Overlap index for (merged) ExpressionSet*

Description

Measures the expected overlap of samples, the higher this overlap, the better the integration process.

Usage

```
measureSamplesOverlap(eset, batchAnnot="Batch")
```

Arguments

eset	ExpressionSet object.
batchAnnot	Column in pData(eset) containing the batch information.

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Calculate Samples Overlap index for both datasets
measureSamplesOverlap(eset_NONE, batchAnnot="Study");
measureSamplesOverlap(eset_COMBAT, batchAnnot="Study");
```

`measureSignificantGenesOverlap`

Compares the overlap in significant genes between two independent studies before and after applying batch effect removal methods.

Description

The quality of batch effect removal is proportional to the number of significant differentially expressed genes (DEGs) found in the newly combined study which are also found in all of the individual studies to be combined.

Usage

```
measureSignificantGenesOverlap(esetBEFORE, esetAFTER, batchAnnot="Batch", targetAnnot)
```

Arguments

`esetBEFORE`	ExpressionSet of combined study before applying batch removal method.
`esetAFTER`	ExpressionSet of combined study after applying batch removal method.
`batchAnnot`	Column in pData(eset) containing the individual batch information.
`targetAnnot`	Column in pData(eset) containing the biological variable of interest for finding differentially expressed genes.

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
```

```
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Calculate significant genes overlap index for both datasets
measureSignificantGenesOverlap(eset_NONE, eset_COMBAT, batchAnnot="Study", targetAnnot="Disease");
```

`merge` *General method to merge different ExpressionSets*

Description

General method to merge different ExpressionSets by applying different techniques to remove batch effects.

Usage

```
merge(esets, method="NONE");
```

Arguments

`esets`	List of ExpressionSet objects.
`method`	Merging method aimed at removing batch effects. Possible options are: BMC, COMBAT, DWD, GENENORM, GENESHIFT, NONE and XPN. 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.

'GENESHIFT': GENESHIFT is a new non-parametric batch effect removal method [5] based on two key elements from statistics: empirical density estimation and the inner product as a distance measure between two probability density functions. As principle, GENESHIFT removes the batch effect gene-wise, by finding the best match between the distribution of the corresponding genes in the two studies to be merged. One dataset is considered as being reference while the other one is shifted gene-wise with an offset estimated based on the best gene matching.

'NONE': 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.

[5] Manuscript in preparation. For more information contact: vlazar@vub.ac.be.

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using different methods:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_BMC = merge(esets, method="BMC");
```

plotDendrogram *Create dendrogram from (merged) ExpressionSet*

Description

Create dendrogram from ExpressionSet after performing hierachical clustering.

Usage

```
plotDendrogram(eset, batchAnnot="Batch", legend=TRUE, file=NULL, ...)
```

Arguments

eset	ExpressionSet object.
batchAnnot	Column in pData(eset) containing the batch information.
legend	If TRUE a legend will be provided next to the dendrogram 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("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Visual inspection of the two merged datasets through an MDS plot
plotDendrogram(eset_NONE, batchAnnot="Study");
plotDendrogram(eset_COMBAT, batchAnnot="Study");
```

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

Description

Gene-wise boxplots describe the gene-wise distribution of samples. Samples can be grouped together using the batchAnnot parameter and can be colored usig the targetAnnot parameter for optimal visualization of the possible batch effects.

Usage

```
plotGeneWiseBoxPlot(eset, batchAnnot="Batch", targetAnnot=NULL, gene=NULL, legend=TRUE, file=NUL
```

Arguments

eset	ExpressionSet object.
targetAnnot	Column in pData(eset) containing the batch information.
batchAnnot	Column in pData(eset) containing the biological variable of interest.
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("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Visual inspection of a random gene in the two merged datasets
gene = sample(rownames(exprs(eset_NONE)), 1);
plotGeneWiseBoxPlot(eset_NONE, batchAnnot="Study", targetAnnot="Disease", gene=gene);
plotGeneWiseBoxPlot(eset_COMBAT, batchAnnot="Study", targetAnnot="Disease", gene=gene);
```

plotGeneWiseDensity *Create gene-wise density plots from (merged) ExpressionSet*

Description

Gene-wise density plots describe the distribution of genes across different samples.

Usage

```
plotGeneWiseDensity(eset, batchAnnot="Batch", gene=NULL, legend=TRUE, file=NULL, main=NULL, ...)
```

Arguments

eset	ExpressionSet object.
batchAnnot	Column in pData(eset) containing the individual batch information.
gene	Gene for which the density plots 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 density plot.
file	If defined, the resulting plot will be stored as a pdf file instead of shown interactively.
main	If defined, a title for the plot can be defined. If not defined the gene name will be selected.
...	Additional parameters for the 'plot' function (e.g. 'main').

Examples

```
# retrieve two datasets:
library(inSilicoDb);
eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Visual inspection of a random gene in the two merged datasets
gene = sample(rownames(exprs(eset_NONE)), 1);
plotGeneWiseDensity(eset_NONE, batchAnnot="Study", gene=gene);
plotGeneWiseDensity(eset_COMBAT, batchAnnot="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, batchAnnot="Batch", targetAnnot=NULL, legend=TRUE, file=NULL, ...)
```

Arguments

eset	ExpressionSet object.
batchAnnot	Column in pData(eset) containing the individual batch information. All samples with the same value in pData(eset)[,batchAnnot] will share the same symbol.
targetAnnot	Column in pData(eset) containing the biological variable of interest. All samples with the same value in pData(eset)[,targetAnnot] will share the same color.
legend	If TRUE a legend will be provided next to the MDS plot for both batchAnnot and targetAnnot.
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("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Visual inspection of the two merged datasets through an MDS plot
plotMDS(eset_NONE, batchAnnot="Study", targetAnnot="Disease");
plotMDS(eset_COMBAT, batchAnnot="Study", targetAnnot="Disease");
```

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, batchAnnot="Batch", legend=TRUE, file=NULL, ...)
```

Arguments

eset	ExpressionSet object.
batchAnnot	Column in pData(eset) containing the individual batch information.
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("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
esets = list(eset1,eset2);

# merge them using the 'NONE' and 'COMBAT' method:
library(inSilicoMerging);
eset_NONE = merge(esets, method="NONE");
eset_COMBAT = merge(esets, method="COMBAT");

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

# Visual inspection of the two merged datasets through an RLE plot
plotRLE(eset_NONE, batchAnnot="Study")
plotRLE(eset_COMBAT, batchAnnot="Study")
```

Index