Using the inSilicoMerging package

Jonatan Taminau*, Stijn Meganck, Cosmin Lazar

CoMo, Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussels, Belgium

1 Merging Gene Expression Data

An increasing amount of gene expression datasets is available through public repositories like for example GEO [2] and ArrayExpress [6]. Combining such data from different studies could be beneficial for the discovery of new biological insights and could increase the statistical power of gene expression analysis. However, the use of different experimentation plans, platforms and methodologies by different research groups introduces undesired batch effects in the gene expression values. This problem hinders and complicates further analysis and can even lead to incorrect conclusions [3]. Several methods to remove the batch effect but at the same time to preserve the biological variance inside the data are proposed in the last years. The inSilicoMerging package combines several of the most used methods to remove these unwanted batch effects for integrative analysis or large-scale analysis. All methods are implemented in such a way that they can be consistently used inside the Bioconductor framework.

2 Using the inSilicoMerging package

Using the inSilicoMerging package is straightforward since it mainly involves only a single function:

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

with `esets` a list of ExpressionSet objects and `method` one of the following options: `BMC`, `COMBAT`, `DWD`, `GENENORM`, `NONE` and `XPN`. Each of those methods is already extensively reported in literature but is nevertheless briefly explained in the following section.

*jtaminau@vub.ac.be

2.1 Evaluation through Visualization Tools

In order to visually inspect a merged dataset to have some direct feedback on its effect, five different visual validation methods are provided:

```
> plotMDS(eset, ...)
> plotRLE(eset, ...)
> plotDendrogram(eset, ...)
> plotGeneWiseBoxPlot(eset, ...)
> plotGeneWiseDensity(eset, ...)
```

`plotMDS` creates a *double-labeled* Multidimensional Scaling (MDS) plot. In this plot, all samples can be labeled by color and by symbol. This might be useful since for each sample its biological phenotype of interest and the study it originates from can be visualized simultaneously, giving an indication of the effectiveness of the used merging method. `plotRLE` creates a relative log expression (RLE) plot, which was initially proposed to measure the overall quality of a dataset but can also be used in this context. `plotDendrogram` creates a dendrogram after applying hierarchical clustering. Finally, `plotGeneWiseBoxPlot` and `plotGeneWiseDensity` provide a local visualization by looking at the boxplots/densities of genes across samples. All these methods are illustrated in the examples section.

2.2 Evaluation through Quantitative Measures

In addition of the five visual inspection methods the `inSilicoMerging` package also provides six different quantitative evaluation measures:

```
> measureAsymetry(eset, ...)
> measureGenesOverlap(eset, ...)
> measureSamplesOverlap(eset, ...)
> measureGenesMeanCorrCoef(eset1, eset2, ...)
> measureSamplesMeanCorrCoef(eset1, eset2, ...)
> measureSignificantGenesOverlap(eset1, eset2, ...)
```

`measureAsymetry` compares the distribution of samples asymmetry, quantified by the *skewness*, before and after batch removal. This index should have a value close to 0. `measureGenesOverlap` and `measureSamplesOverlap` measure the expected overlap of genes and samples between two independent studies, the higher this overlap, the better the integration process. `measureGenesMeanCorrCoef` and `measureSamplesMeanCorrCoef` both indicate how the data is effected after batch removal. The method that least affects the data should be preferred. Note that this evaluation method does not give any clues on how effective a method is. The idea of the `measureSignificantGenesOverlap` method is that 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.

3 Different Merging Methods

Below we list, alphabetically, the merging techniques implemented in this package. Note that after using any of those methods the resulting merged dataset only contains the *common* list of genes/probes between all studies.

BMC

In [8] 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:

$$\hat{x}_{ij}^k = x_{ij}^k - \overline{x}_i^k \tag{1}$$

This technique was proposed to eliminate multiplicative bias.

COMBAT

Empirical Bayes [4] 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. It is assumed that measured gene expression values of gene i in sample j of batch k can be expressed as:

$$x_{ij}^k = \alpha_i + \mathbf{C}\beta_i + \gamma_i^k + \delta_i^k \epsilon_{ij}^k \tag{2}$$

where α_i is the overall gene expression, $\mathbf{C}$ is a design matrix for sample conditions, β_i is the vector of regression coefficients corresponding to $\mathbf{X}$, γ_i^k and δ_i^k are the additive and multiplicative batch effects for gene i in batch k respectively and ϵ_{ij}^k are error terms.

DWD

By searching for the separating hyperplane between data coming from different batches, Distance-weighted discrimination (DWD), an adaptation of Support Vector Machines (SVM), allows to remove bias by projecting the different batches on the hyperplane, calculating the batch mean $\overline{b}$ distance to the hyperplane and then subtracting the normal vector Δ of this plane multiplied by the mean [1].

$$\hat{x}_{ij}^k = x_{ij}^k - \overline{b}\Delta \tag{3}$$

GENENORM

One of the simplest mathematical transformations to make datasets more comparable is z-score normalization. In this method, for each gene expression value x_{ij} in each study separately all values are modified by subtracting the mean $\overline{x}_i$ of the gene in that dataset divided by its standard deviation σ_i:

$$\hat{x}_{ij}^k = \frac{x_{ij}^k - \overline{x}_i^k}{\sigma_{i.}^k} \tag{4}$$

NONE

The most basic approach to combine two datasets is to simply *paste* them together without any transformation. This can be used as a baseline against which other techniques can be compared.

XPN

The basic idea behind the cross-platform normalization [7] approach is to identify homogeneous blocks (clusters) of gene 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. For a platform k, gene i and sample j, the recorded gene expression is given by:

$$x_{ij}^k = A_{\alpha^*(i),\beta_k^*(j)}^k b_i^k + c_i^k + \sigma_i^k \epsilon_{ij}^k \tag{5}$$

where A_{α^*,β^*}^k is a block mean and b_i^k and c_i^k represent gene and platform specific sensitivity and offset parameters respectively. The functions $\alpha^*()$ and $\beta^*()$ map a specific gene measurement in a sample to their corresponding multi-platform cluster. The noise variables ϵ_{ij}^k are assumed independent standard gaussians. `XPN` uses an iterative scheme to update the parameters in Equation 5 until convergence to a local minimum, giving:

$$\hat{x}_{ij}^k = \hat{A}_{\alpha^*(i),\beta_k^*(j)}^k \hat{b}_i^k + \hat{c}_i^k + \hat{\sigma}_i^k \hat{\epsilon}_{ij}^k \tag{6}$$

More details can be found in [7].

3.1 Merging two-by-two

Some merging technique are only reported and implemented to merge exactly two studies (e.g. `XPN` [7] and `DWD` [1]). In order to be able to merge any number of studies, this package added an additional step. This step combines all studies two-by-two and is called recursively on the intermediate results until only one, merged, dataset remains. Its behavior is illustrated in the following example:

```
list of studies = [ A ; B ; C ; D ; E ]
m(X,Y) = applying merging technique 'm' on dataset 'X' and 'Y'
combineByTwo:
  iteration 1 : [ E ; m(A,B) ; m(C,D) ]   => [ E ; AB ; CD ]
  iteration 2 : [ CD ; m(E,AB) ]          => [ CD ; EAB ]
  iteration 3 : [ m(CD,EAB) ]             => [ CDEAB ]
```

4 Example

For this example we retrieve two Lung Cancer datasets using the inSilicoDb package [9]. Both datasets were assayed on different platforms (Affymetrix Human Genome U133A Array and Affymetrix Human Genome U133 Plus 2.0 Array) and then preprocessed using fRMA [5].

```
> library(inSilicoDb)
> eset1 = getDataset("GSE19804", "GPL570", norm="FRMA", genes=TRUE);
> eset2 = getDataset("GSE10072", "GPL96",  norm="FRMA", genes=TRUE);
> esets = list(eset1,eset2);
```

Both studies contain normal and tumor samples and are already consistently annotated with a common `Disease` feature:

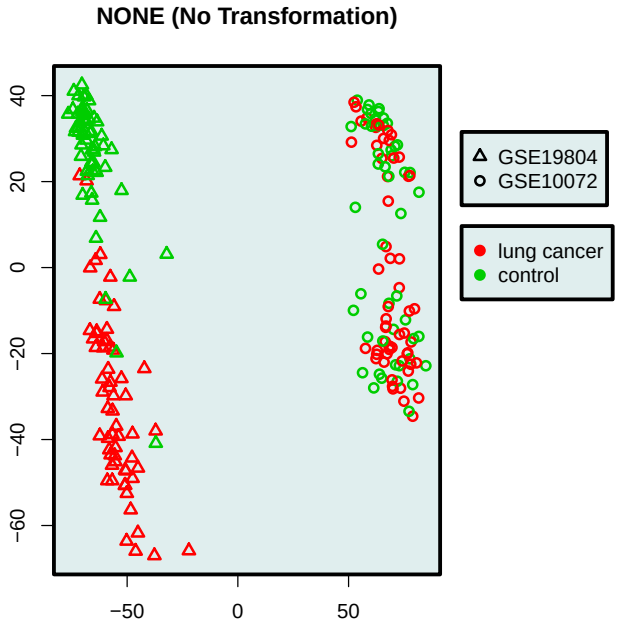
```
> table(pData(eset1)[,"Disease"]);

    control lung cancer
         60          60

> table(pData(eset2)[,"Disease"]);

    control lung cancer
         49          58
```

We now can simply merge both studies without applying any transformation:

```
> library(inSilicoMerging);
> eset_NONE = merge(esets, method="NONE");
```

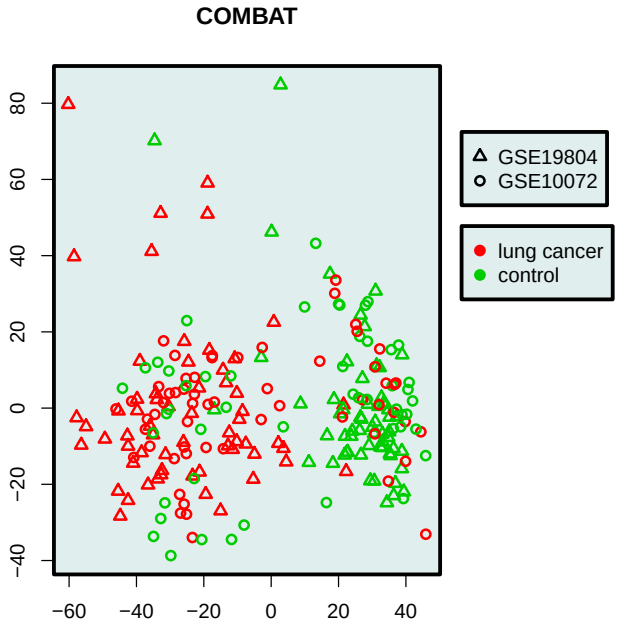
To further investigate the combined data we can use the `plotMDS` function to have a first visual inspection.

```
> plotMDS(eset_NONE,
+         targetAnnot="Disease",
+         batchAnnot="Study",
+         main="NONE (No Transformation)");
```



From this plot we can immediately notice a very strong dataset-bias (probably due to the difference in platform) while we would expect that all control samples from both studies would cluster together. Let us try another method to see if we can solve this issue:

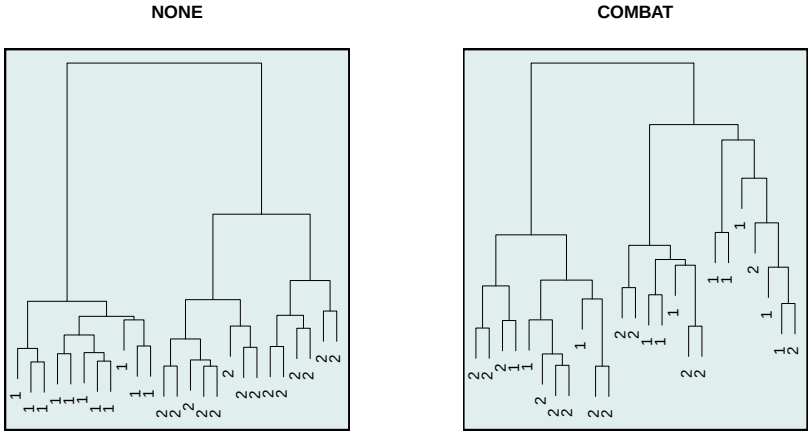
```
> eset_COMBAT = merge(esets, method="COMBAT");
> plotMDS(eset_COMBAT,
+         targetAnnot="Disease",
+         batchAnnot="Study",
+         main="COMBAT");
```



This clearly looks better. Both studies are mixed together and the biological phenotype of interest (tumor versus normal) is preserved in the merged dataset.

In a similar way we can use the other visualization methods too. Another method to look at the distances between samples is for example looking at the result of (hierarchical) clustering. This is provided by the `plotDendrogram` function:

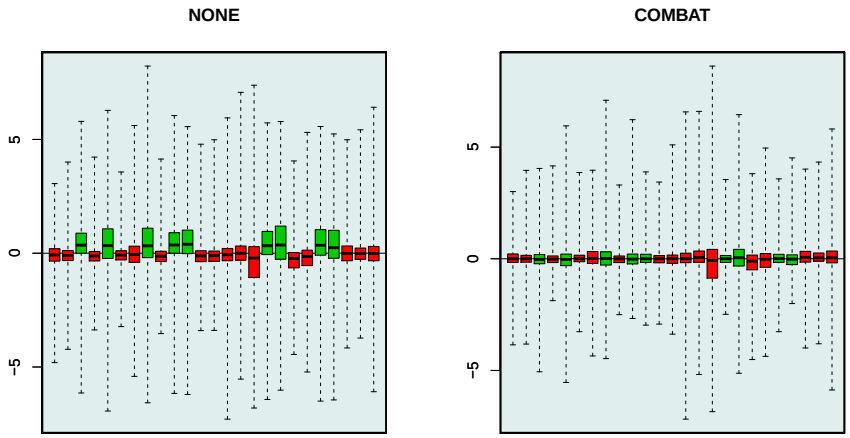
```
> par(mfrow=c(1,2))
> select = sample(1:ncol(eset_NONE),25);
> plotDendrogram(eset_NONE[,select],
+                batchAnnot="Study",
+                legend=FALSE,
+                main="NONE");
> plotDendrogram(eset_COMBAT[,select],
+                batchAnnot="Study",
+                legend=FALSE,
+                main="COMBAT");
```



For clarity purposes we only select 25 (random) samples. The different samples are labeled, also for clarity purposes, by numbers corresponding to the study they originate from instead of their full sample name. We can compare the merging without transformation (`NONE`) on the left and after using the `COMBAT` method on the right.

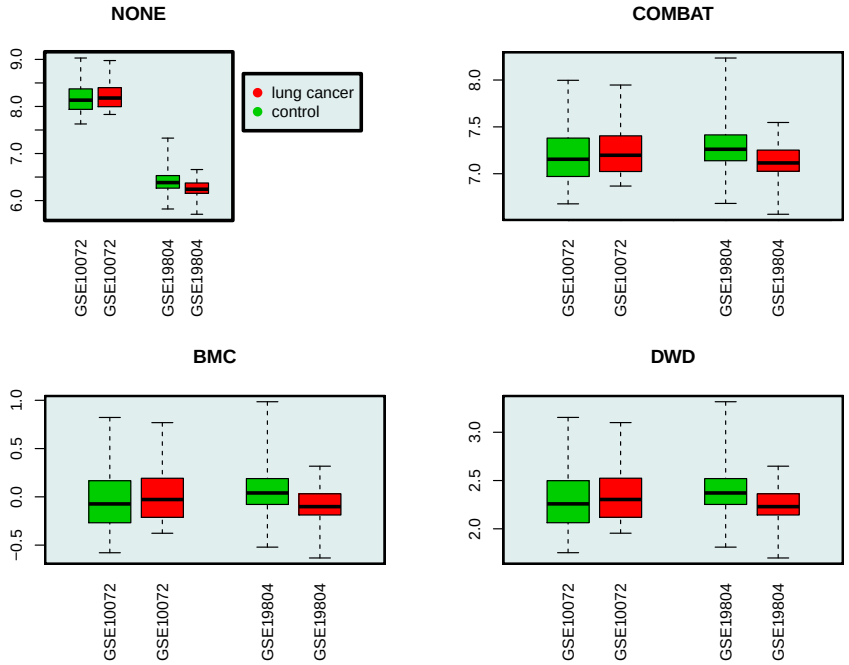
To illustrate the RLE plots we again only select 25 samples. In this plot the samples are colored based on the study they originate from.

```
> par(mfrow=c(1,2))
> select = sample(1:ncol(eset_NONE),25);
> plotRLE(eset_NONE[,select],
+         batchAnnot="Study",
+         legend=FALSE,
+         main="NONE");
> plotRLE(eset_COMBAT[,select],
+         batchAnnot="Study",
+         legend=FALSE,
+         main="COMBAT");
```

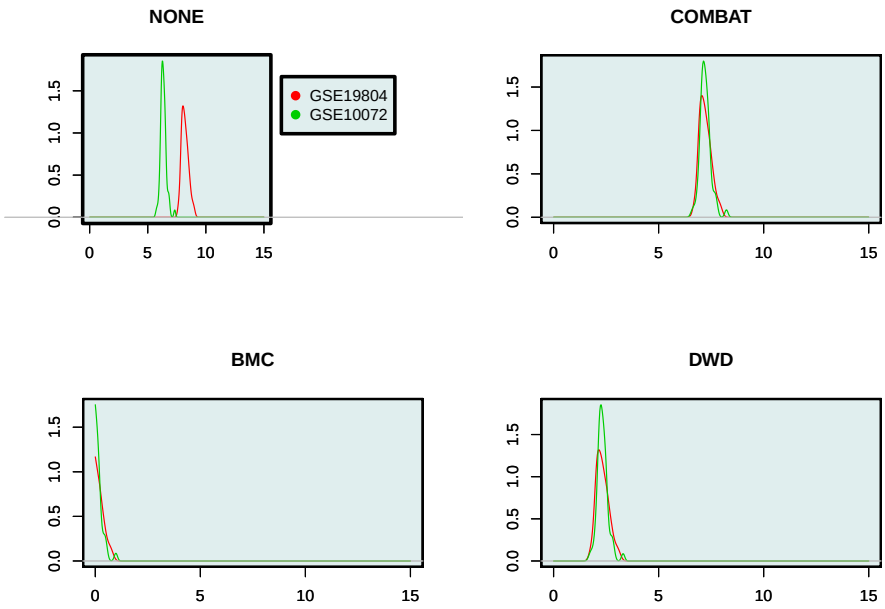


In contrast to the three previous methods which illustrated the global bias between two datasets we also can obtain a very local view this time. With the following two functions, the local effect of each method on the gene level can also be illustrated. This effect is of course different for every gene. We arbitrary select the MYL4 gene to illustrate the batch effects through gene-wise box plots and through gene density plots.

```
> eset_BMC = merge(esets, method="BMC");
> eset_DWD = merge(esets, method="DWD");
> gene = "MYL4";
> par(mfrow=c(2,2));
> plotGeneWiseBoxPlot(eset_NONE, targetAnnot ="Disease", batchAnnot ="Study",
+                     gene=gene, legend=TRUE, main="NONE");
> plotGeneWiseBoxPlot(eset_COMBAT, targetAnnot ="Disease", batchAnnot ="Study",
+                     gene=gene, legend=FALSE, main="COMBAT");
> plotGeneWiseBoxPlot(eset_BMC, targetAnnot ="Disease", batchAnnot ="Study",
+                     gene=gene, legend=FALSE, main="BMC");
> plotGeneWiseBoxPlot(eset_DWD, targetAnnot ="Disease", batchAnnot ="Study",
+                     gene=gene, legend=FALSE, main="DWD");
```




```
> gene = "MYL4";
> par(mfrow=c(2,2));
> plotGeneWiseDensity(eset_NONE, batchAnnot ="Study",
+                                          gene=gene, legend=TRUE, main="NONE");
> plotGeneWiseDensity(eset_COMBAT, batchAnnot ="Study",
+                                          gene=gene, legend=FALSE, main="COMBAT");
> plotGeneWiseDensity(eset_BMC, batchAnnot ="Study",
+                                          gene=gene, legend=FALSE, main="BMC");
> plotGeneWiseDensity(eset_DWD, batchAnnot ="Study",
+                                          gene=gene, legend=FALSE, main="DWD");
```



5 Conclusion

As this example illustrates, it is now straightforward to merge a number of gene expression studies by applying different existing methods. A number of simple visualization tools are provided for a first inspection of the merged dataset(s). In a similar way the quantitative measures can be applied as well.

6 Session Info

```
> sessionInfo()

R version 2.15.1 (2012-06-22)
Platform: i386-apple-darwin9.8.0/i386 (32-bit)

locale:
[1] C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] inSilicoMerging_1.0.8 limma_3.12.1          DWD_0.10
```

```
[4] Matrix_1.0-7          lattice_0.20-10       inSilicoDb_1.4.0
[7] Biobase_2.16.0        BiocGenerics_0.2.0    rjson_0.2.9

loaded via a namespace (and not attached):
[1] RCurl_1.91-1 amap_0.8-7   grid_2.15.1  moments_0.13 tools_2.15.1
```

References

[1] Monica Benito, Joel Parker, Quan Du, Junyuan Wu, Dong Xiang, Charles M. Perou, and J. S. Marron. Adjustment of systematic microarray data biases. *Bioinformatics*, 20(1):105–114, 2004.

[2] Ron Edgar, Michael Domrachev, and Alex E Lash. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res*, 30(1):207–10, Jan 2002.

[3] Jeffrey T Leek, Robert B Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W Evan Johnson, Donald Geman, Keith Baggerly, and Rafael A Irizarry. Tackling the widespread and critical impact of batch effects in high-throughput data. *Nat Rev Genet*, 11(10):733–9, 2010.

[4] Cheng Li and Ariel Rabinovic. Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, 8(1):118–127, 2007.

[5] Matthew N McCall, Benjamin M Bolstad, and Rafael A Irizarry. Frozen robust multiarray analysis (fRMA). *Biostatistics*, 11(2):242–53, 2010.

[6] Helen Parkinson, Ugis Sarkans, Nikolay Kolesnikov, Niran Abeygunawardena, Tony Burdett, Miroslaw Dylag, Ibrahim Emam, Anna Farne, Emma Hastings, Ele Holloway, Natalja Kurbatova, Margus Lukk, James Malone, Roby Mani, Ekaterina Pilicheva, Gabriella Rustici, Anjan Sharma, Eleanor Williams, Tomasz Adamusiak, Marco Brandizi, Nataliya Sklyar, and Alvis Brazma. Arrayexpress update–an archive of microarray and high-throughput sequencing-based functional genomics experiments. *Nucleic Acids Res*, 39(Database issue):D1002–4, Jan 2011.

[7] Andrey A. Shabalin, Håkon Tjelmeland, Cheng Fan, Charles M. Perou, and Andrew B. Nobel. Merging two gene-expression studies via cross-platform normalization. *Bioinformatics*, 24(9):1154–1160, 2008.

[8] Andrew Sims, Graeme Smethurst, Yvonne Hey, Michal Okoniewski, Stuart Pepper, Anthony Howell, Crispin Miller, and Robert Clarke. The removal of multiplicative, systematic bias allows integration of breast cancer gene expression datasets - improving meta-analysis and prediction of prognosis. *BMC Medical Genomics*, 1(1):42, 2008.

[9] Jonatan Taminau, David Steenhoff, Alain Coletta, Stijn Meganck, Cosmin Lazar, Virginie de Schaetzen, Robin Duque, Colin Molter, Hugues Bersini, Ann Nowé, and David Y. Weiss Solís. inSilicoDb: an R/Bioconductor package for accessing human Affymetrix expert-curated datasets from GEO. *Bioinformatics*, 27(22):3204–3205, 2011.