Comparison of location-scale and matrix factorization batch effect removal methods on gene expression datasets

Emilie Renard
ICTEAM Institute
Université catholique de Louvain
Louvain-la-Neuve, 1348, Belgium
Email: emilie.renard@uclouvain.be

P.-A. Absil
ICTEAM Institute
Université catholique de Louvain
Louvain-la-Neuve, 1348, Belgium
Email: pa.absil@uclouvain.be

Tech. report UCL-INMA-2017.09-v1 – 2017-10-24
http://sites.uclouvain.be/absil/2017.09

***Abstract*—Merging gene expression datasets is a simple way to increase the number of samples in an analysis. However experimental and data processing conditions, which are proper to each dataset or batch, generally influence the expression values and can hide the biological effect of interest. It is then important to normalize the bigger merged dataset, as failing to adjust for those batch effects may adversely impact statistical inference. Batch effect removal methods are generally based on a location-scale approach, however less widespread methods based on matrix factorization have also been proposed. We investigate on breast cancer data how those batch effect removal methods improve (or possibly degrade) the performance of simple classifiers. Our results indicate that the matrix factorization approach would deserve greater attention, as it gives results at least as good as common location-scale methods, and even significantly better results in specific cases.**

I. Introduction

Nowadays, the development of sequencing technologies allows to measure gene expression levels at a reasonable cost. The analysis of the resulting data helps to better understand how genes are working, with the goal of developing better cures for genetic diseases such as cancer. Due to different constraints such as the limited number of samples that can be processed at the same time in an experiment, the size of such datasets is often limited in samples. However, statistical inferences need a high number of samples to be robust enough and generalizable to other data. As more and more of those datasets are available on public repositories such as GEO http://www.ncbi.nlm.nih.gov/geo/, merging and combining different datasets appears as a simple solution to increase the number of samples analyzed and potentially improve the relevance of the biological information extracted.

Expression levels of genes are the result of interactions between different biological processes. When measuring those expression levels, noise may also be added at each step of data acquisition due to imprecisions. In particular, different biases can be introduced depending on experimental conditions. Such confounding factors, or batch effects, that complicate the analysis of genomic data can be for example due to difference in chip type or platform, procedures that can differ from one laboratory to another, storage conditions, ambient conditions during preparation, etc. A carefully designed experimental process can limit the impact of such effects, but some are often unavoidable, especially when a large number of samples is necessary. Those batch effects can be quite large and hide the effects related to the biological process of interest. Not including those effects in the analysis process may adversely affect the validity of biological conclusions drawn from the datasets [1]–[3]. It is then important to be able to combine data from different sources while removing the batch effects. The difficulty is that the precise effects of those technical artefacts on gene expression levels is often unknown. However some partial information is usually available, such as the batch number or the date of experiment, and can be used as a proxy for those effects.

Available batch effect removal methods can be classified in two main approaches: location-scale methods and matrix factorization methods. The location-scale methods assume a model for the data distribution within batches, and adjust the data within each batch to fit this model. This approach is the most straight-forward one and many methods have already been proposed: XPN [4], DWD [5], ratio-based methods [6], ComBat [7], quantile based methods [8], mean or median centering [9], [10], etc. The matrix factorization based methods assume that the gene-by-sample expression matrix can be represented by a small set of rank-one components which can be estimated by means of matrix factorization. The components that correlate with the batch number are then removed to obtain the normalized dataset [11], [12]. In [2], [3], matrix factorization is used to model covariates in a differentially expressed gene (DEG) detection process. Matrix factorization based methods are less used, probably because of the indirect approach to the problem: if no clear batch effect is recovered in the rank-one components, then the data matrix is left unchanged. However, not forcing the direct removal of differences between batches can be an advantage as it is more adaptable to cases where real technical artifacts are not exactly linked to the known batches. Another option is to look directly

for patterns present in all batches (or latent variables). Various implementation of such an approach were already proposed in the case of multi-omic data integration (combining data where the common dimension is the samples), but a recent adaptation was proposed in [13] in the specific context of classification based on independent datasets.

The presence or absence of batch effects in a dataset—and thus the effectiveness of batch effect removal methods—can be evaluated by different methods (see, e.g., [14] for an overview), which can be classified in three main groups: local approaches, global approaches and, in supervised cases, performance based approaches. Global methods aim to illustrate the global behavior of the dataset: the evaluation of batch effect presence uses many features at once. For example, the global behavior of all genes can be summarized by a clustering dendogram or a plot of the first principal components. A clustering of samples by batch shows presence of batch effects; it means that the predominant trend present in the data is linked to batches. However it does not imply that all genes are affected to the same extent, or even affected at all.

On the contrary, local evaluation methods examine the behavior of one gene at a time: expression levels of a gene should have the same behavior (typically similar probability distributions) for all batches. When evaluating a batch effect removal method, the clustering by batch and/or the differences in behavior across batches should disappear (or at least be weaker).

Those evaluation methods are unsupervised in the sense that there is no ground truth to refer to. Hence they are unable to tell if the evaluated batch effect removal method has removed differences between batches that are due to technical artifacts (as they should) or to biological differences between batches (as they should not). Batch effect removal methods that also remove differences between batches that have biological origins—a major danger when the batches are imbalanced—can thus be favored by such unsupervised evaluation methods.

Now, if there is some ground truth to refer to, then a more meaningful evaluation of batch effect removal methods can be carried out. If some genes are known to be truly (not) differentially expressed, the proportion of those genes in the DEG list after removal of batch effect should ideally be higher (lower). P-values corresponding to null genes or negative control genes should be uniformly distributed across $[0, 1]$. The list of DEG should also be more stable from one batch to another. In prediction tasks where the final objective is for example to determine to which class a new sample belongs, performances should be improved after batch effect removal. See for example [14] for a list of techniques to evaluate batch effects presence.

Various comparisons of different batch effect removal methods can be found in [4], [6], [14]–[18]. While some methods appear more often among the best ones, there is not one best method outstanding from those studies as the context and objective of comparison varies. The set of tested methods varies from one study to another, but very few matrix factorization based methods were considered in those comparisons. Reference [12] compared some matrix factorization methods to ComBat in a classification task and showed that such kind of approach can improve results. However, only four batch effect removal methods were compared, only one classifier was considered, and training and testing sets were separated only after removal of batch effects.

In this paper, we carry out a more thorough investigation of the impact of general purpose batch effect removal methods on sample classification tasks. We compare the two families of approaches (location-scale vs matrix factorization) to understand their advantages and weaknesses. For this we use nine different methods (15 with variants) and five classifiers, taking care of separating training and testing data from the very beginning (i.e., as an addon method such as explained in [18]). Our salient finding is that some matrix factorization methods, though less widely investigated in the literature, have the edge over location-scale methods; see Section IV for details.

The paper is organized as follows. Section II details the methods examined, which are experimented and analyzed in Section III, and conclusions are drawn in Section IV.

II. Batch effect removal methods

Among the many batch effect removal methods mentioned above, we choose to investigate various well known location-scale methods and a few matrix factorization based ones. The choice of those specific methods was quite arbitrary, however we try to represent the different cases. Some methods, such as standardization, are really simple while others, such as ComBat, are more complex. FAbatch combines location-scale and matrix factorization approaches, and rgCCA is to our knowledge tested for the first time in this context. More in depth explanations of the chosen methods are detailed in this Section.

A. Location-scale methods

Location-scale methods are maybe the most intuitive way to handle the data. Choosing a reasonable model representing the probability distribution of gene expression, and assuming that genes behave in the same way in each batch, expression values are adapted to fit this model within each batch. The goal is to let each gene have a similar mean and/or variance in each batch. A main hypothesis in such methods is that by adjusting the gene distributions no biological information is removed, and that each batch groups samples with reasonably similar experimental conditions.

The simplest way to normalize a dataset in order to remove batch effects is to center (called $Centering$) or standardize (called Std) each batch separately. That is, for each gene in each batch, the expression values are centered, and divided by their standard deviation if desired.

Other simple approaches are ratio-based methods [6], which scale each value with a reference. Usual references can be arithmetic (called $RatioA$) or geometric (called $RatioG$) mean value of the corresponding variable in the same batch.

A widely used and more complex location-scale method is ComBat [7] (called $ComBat$). The expression value of gene i for sample j in batch b is modeled as

$$X_{bij} = \alpha_i + \beta_i C_j + \gamma_{bi} + \delta_{bi}\epsilon_{bij}$$

where α_i is the overall gene expression, and C_j is the vector of known covariates representing the sample conditions (such as batch membership). The error term ϵ_{bij} is assumed to follow a normal distribution $N(0, \sigma_i^2)$. Additive and multiplicative batch effects are represented by parameters γ_{bi} and δ_{bi}. ComBat uses a Bayesian approach to model the different parameters, and then removes the batch effects from the data to obtain the clean data $X^*_{bij} = \hat{\epsilon}_{bij} + \hat{\alpha}_i + \hat{\beta}_i C_j$. By pooling information across genes, this approach is more robust to outliers in small sample sizes.

B. Matrix factorization based methods

The main assumption is that the gene-by-sample matrix X can be represented using a low-rank factorization :

$$X \approx AB^T = \sum_k A(:,k)B(:,k)^T. \quad (1)$$

$A(:,k)$ can be interpreted as the gene activation pattern of component k and $B(:,k)$ as the weights of this pattern in the samples. For each k, the highly activated genes in $A(:,k)$ can be viewed as a group of genes working together in a specific condition, and $B(:,k)$ is the intensity of this condition among samples. The hope is then that some of the components represent the biological conditions of interest (and should be kept) while others represent the batch effects (and could be removed from the data). In other words, we are looking for a basis $U \subset A$ representing the gene activation patterns not linked to batch. This basis U can then be used in an "addon" way as described in Subsection III-B.

When dealing with different batches, two approaches can be investigated. The first one is to look directly for components present in all batches (or latent variables) such that $X_b \approx UV_b^T$ (where b represents the batch) . The other option is to directly apply the factorization on X, the concatenation of the m batches, to obtain $X = [X_1...X_m] \approx AB^T = A[B_1...B_m]^T$ and then remove components correlating with batch number. Specific supervised methods were also developed to integrate class labels in the process.

1) Keeping common components: The first approach is inspired by Canonical Correlation Analysis [19], which aims to find linear combinations of the variables of two datasets with a maximum correlation. A generalization to more than two datasets was proposed by [20]:

$$\max_{t_b} \sum_{j \neq k} c_{jk} g(Cov(X_j t_j, X_k t_k))$$
$$\text{s.t. } \tau_b ||t_b||^2 + (1-\tau_b) Var(X_b t_b) = 1 \quad \forall b$$

where in our case X_b represents the matrix of expression values of batch b. Function g is usually the identity (the method is then called $rgCCA_{id}$), the absolute value or the square function (the method is then called $rgCCA_{sq}$). Parameter c_{jk} represents the link between batches j and k, and $0 \leq \tau_b \leq 1$ let us choose between the type of constraint on t_b and/or $X_b t_b$. We took all $c_{jk} = 1$ and $\tau = 1$, common basis U was then obtained as $U = XV(V^TV)^{-1}$ with $V = [t_1...t_m]$.

2) Removing batch related components: A template for the second approach, adapted from [12], is detailed in Table I. The first step is to factorize the matrix X (line 1). Many methods exist to factorize a matrix, depending on the properties the factorization components have to fulfill. Imposing orthogonality among components leads to a Singular Value Decomposition. Analysis of gene expression data using SVD was first proposed in [11]. Minimizing statistical dependence across component leads to Independent Component Analysis (ICA) methods. Different variants exist for such methods depending on how statistical dependence is evaluated, and wether we want to impose independence among genes or samples dimension, or even using a trade-off between both options [12]. ICA was shown to better model the different sources of variation than SVD [3]. Other factorization methods exist in the literature, such as Non-negative Matrix Factorization [21] which can be used when dealing with non-negative matrix values to obtain components with non-negative values only.

Once the factorization is computed, we select the $B_{:,k}$'s that correlate with the batch. If a component presents enough correlation with the batch (line 3), then this component is selected. As batch is a categorical information and the $B_{:,k}$'s are continuous, the usual linear correlation formula (Pearson or Spearman) cannot be used. To estimate which components are related to batch, as in [12] we use the R^2 value that measures how well a variable x (here, c) can predict a variable y (here, $B_{:k}$) in a linear model:

$$R^2(x,y) \equiv 1 - \frac{SS_{res}}{SS_{tot}}.$$

$SS_{tot} = \sum_i (y_i - \bar{y})^2$ is the sum of squares of the prediction errors if we take the mean $\bar{y} = \frac{1}{n}\sum_{i=1}^n y_i$ as predictor of y. $SS_{res} = \sum_i (y_i - \hat{y}_i)^2$ is the sum of squares of the prediction errors if we use a linear model $\hat{y}_i = f(x_i)$ as predictor: if x is continuous the prediction model is a linear regression, if x is categorical we use a class mean. The R^2 value indicates the proportion of the variance in y that can be predicted from x, and has the advantage to be usable with categorical or continuous variables. So the higher the R^2 value, the better the association between both variables. As the batch information is categorical, $R^2(c, B_{:k})$ compares the prediction of B_{ik} by a general mean $\sum_j \frac{B_{jk}}{n}$ or by a batch mean $\sum_{j \in C_i} \frac{B_{jk}}{\#C_i}$ (where C_i represents all samples in the same batch as sample j).

An additional step can be added in the process to check if the selected components do not correlate with some information of interest (lines 4-6, optional). The selected components are then removed from the matrix X to obtain a cleaned dataset (line 7). The common basis U is then generated by the matrix A cleaned from the selected components (line refalgu).

As matrix factorization methods, we tested the singular value decomposition (called SVD) and spatio-temporal ICA

TABLE I
MATRIX FACTORIZATION BASED METHOD
(REMOVING BATCH RELATED COMPONENTS)

Require: X $(p \times n)$ the aggregated dataset to be normalized, c (n) a categorical variable indicating the batch number, $matfact$ the matrix factorization method, $t \in [0, 1]$ the threshold to consider a component associated to c, [optional] c_2 (n) categorical/continuous information that we want to preserve

$A, B \leftarrow \texttt{matfact}(X)$
$R \leftarrow \texttt{cor}(c, B)$
$ix \leftarrow \texttt{which}(R \geq t)$
$R_2 \leftarrow \texttt{cor}(c_2, B)$ {optional}
$ix_2 \leftarrow \texttt{which}(R_2 \geq R)$ {optional}
$ix \leftarrow ix \setminus ix_2$ {optional}
$\hat{X} \leftarrow X - A[:, ix] * B[:, ix]^T$
$U \leftarrow A[:, -ix]$

as presented in [22] with $\alpha \in \{0, 0.25, 0.5, 1\}$ (called $stICA_0$, $stICA_{0.25}$, $stICA_{0.5}$ and $stICA_1$).

3) Supervised methods: A well-known batch effect removal method is surrogate variable analysis [1] (called SVA), which combines SVD and a linear model analysis to estimate the eigengenes from a residual expression matrix from which biological variation has already been removed:

$$x_{ij} = \mu_i + f_i(y_j) + \sum_k \lambda_{ki} g_{kj} + \epsilon_{ij}$$

with $\mu_i + f_i(y_j)$ a linear regression on the phenotype to predict x_{ij} from y_j, and surrogate variables g_{kj} estimated using SVD.

In the same direction, a method combining latent factor adjustment and location-scale was proposed in [23] (called FA):

$$x_{bij} = \alpha_i + \mathbf{a}_{jb}^T \beta_i + \gamma_{bg} + \sum_{l=1}^{k_b} b_{bil} Z_{jbl} + \delta_{bi} \epsilon_{bij}$$

with $Z_{jbl} \sim N(0, 1)$ for all j in $1, ..., k_b$ and $\epsilon_{bij} \sim N(0, \sigma_i^2)$. All known factors of interest, such as the phenotype to predict, are represented in the term $\mathbf{a}_{jb}^T$. Latent variables (including batch effects) are represented by $b_{bil} Z_{jbl}$ and estimated using an EM algorithm.

Those last two batch effect removal methods are supervised in the sense that they explicitly integrate information about the phenotype to predict in their models. It is of course possible to go a step further by computing specifically components correlating with this information, like in [13]. FA and SVA could be seen as weakly supervised in the sense that they try to avoid removing the class information, while [13] aims to find components specifically representing this information and is then more strongly supervised. The method in [13] is then less general than the ones detailed previously, so we did not consider it as a general purpose batch effect removal method.

III. RESULTS AND DISCUSSION

We tested the batch effect removal methods $rgCCA_{id}$, $rgCCA_{sq}$, $stICA_0$, $stICA_{0.25}$, $stICA_{0.5}$ and $stICA_1$ for matrix factorization and $ComBat$, Std, $Centering$, $RatioA$ and $RatioG$ for location-scale. We also tested SVA and FA, two methods using class information, and the case without batch effect removal (called $None$) as reference. We tested all those methods on breast cancer expression; more details on the datasets used can be found in Table II. We took as phenotype of interest to predict the estrogen-receptor status (ER), and consider each dataset with a specific platform as a batch. Features of the different datasets were matched using the common Entrez ID, and samples and features with respectively more than 1% and 10% of missing information were removed, giving an aggregated dataset of 9371 genes and 2209 samples. The missing information was then imputed for each batch using the 'impute' R package. Code is available from https://sites.uclouvain.be/absil/2017.09.

TABLE II
SUMMARY OF BATCHES USED, NAMED AFTER THEIR GSE REFERENCES AND THE PLATFORM USED.

	Batch	Technology	# ER-	# ER+
Train	$GSE2034_{96}$	Affymetrix	77	209
	$GSE21653_{570}$	Affymetrix	113	150
	$GSE2990_{96}$	Affymetrix	34	85
	$GSE17040_{887}$	Agilent	11	44
	$GSE1992_{887}$	Agilent	16	23
	NKI2	Agilent	43	159
Test	$GSE17705_{96}$	Affymetrix	0	298
	$GSE3494_{96}$	Affymetrix	34	213
	$GSE5327_{96}$	Affymetrix	58	0
	$GSE6532_{570}$	Affymetrix	0	87
	$GSE6532_{96}$	Affymetrix	11	115
	$GSE7390_{96}$	Affymetrix	64	134
	$GSE32641_{887}$	Agilent	84	0
	$GSE8465_{887}$	Agilent	17	16
	NKI	Agilent	40	74

A. Global effect of batch effect removal methods

To evaluate the presence of batch effects, we plotted the two first principal components of the datasets (see Figure 1). Principal components represent linear combinations of features (here, genes values) giving the largest possible variance, such that components are uncorrelated: the first components capture most of the variability in the data. On Figure 1, we can see that without removal of batch effect (subplot 'None') the first two principal components of the datasets show a global clustering by batch. Moreover, all Agilent datasets are clustered together on the right. The two batches corresponding to the same platform (GPL570) are also close.

The three other subplots give an idea on how the different types of batch effect removal method work. Clearly, applying a batch effect removal method reduces the batch clustering, and even tends to cluster samples by ER status. Note that by forcing each dataset to have the same mean, ComBat cannot merge the two clusters present in $GSE6532_{96}$. Such distinct clusters are no more present for SVD. However samples belonging to the same batch still tend to stay in the same neighborhood, probably because the batch information is not directly used when computing the new representation of the data. In FA, which combines location-scale and matrix

factorization approaches, there is no more clustering by batch at all.

B. Validation by impact on classification

To compare the different batch effect removal methods, we used them in a whole classification process where the ER status is predicted. The ER status is thus the phenotype to predict, i.e. the outcome. The whole process is described in Table III. The first step is to separate training and testing sets (lines 1–2). Next batch effects are removed from the training datasets with the chosen method (line 3). Optionally, the training labels can be used as extra information to preserve. Here we did not use this option in our experiments except for SVA and FA (in particular, c_2 is not used in the process of Table I). In matrix factorization based methods, we computed the $K = 20$ first components and removed the components with an R^2 value higher than $t = 0.5$.

A basic feature selection is performed by selecting the genes with the best association with the ER label based on a t-test (line 4). A classifier is trained based on those genes (lines 5–6). When training the classifier model, train data are first centered and scaled to ensure to treat all features with the same weight.

Batch effects are then removed from the test dataset X_{te} (line 7) using an addon counterpart of the chosen method. In case of a matrix factorization based approach, in order to comply with the addon requirement [18], the batch effect free dataset $\hat{X}_{te}$ is computed using the projection of X_{te} on the basis U (see Subsection II-B):

$$\hat{X}_{te} = U(U^T U)^{-1} U^T X_{te}.$$

After centering and scaling, the ER labels of testing set are finally predicted using the classifier (line 8).

We repeated lines 5 to 8 using different numbers of selected genes (line 4) and different classifiers (line 6). We started with 2000 features, repeatedly removing 20% of them until reaching less than 150 features; then we removed 10% until a minimum of 10 features. Classifiers tested are k-Nearest Neighboors, Support Vector Machine, Naive Bayes, Random Forest, and Adaboost with logistic regression.

In order to comply with the 'addon' scenario [18] where the model is built once and for all on some studies and then validated on other separate studies, the parameters of batch effect removal methods and classifiers are estimated using only the training datasets. We applied the whole procedure on any subset of four training datasets ($C_6^4 = 15$ experiments), and then tested it on the all nine testing sets (described in Table II).

We choose to use the balanced classification rate (BCR) as performance measure. The balanced classification rate, or balanced accuracy, is computed as $0.5(\frac{TP}{TP+FN} + \frac{TN}{TN+FP})$ and allows to take into account a potential imbalance between classes. The reference value is then 0.5, which corresponds to assigning randomly the labels based on the classes probabilities.

1) Influence of test dataset: As described in Table II, the datasets used vary on different aspects such as the technology used, the number of samples or the proportion of ER+/ER- classes. Figure 2 shows for each dataset and each batch effect removal method the mean BCR obtained when averaging on the different classifiers and training sets. In order to stay concise, results where also averaged on all number of selected features as similar results were obtained when considering different numbers of selected features.

TABLE III
CLASSIFICATION PROCESS

Require: X $(p \times n)$ aggregated matrix of gene expression, y (n) the label to predict, c (n) the batch information
1: $y_{tr}, y_{te} \leftarrow y$
2: $X_{tr}, X_{te} \leftarrow X$
3: $\hat{X}_{tr}, param_{BE} \leftarrow \texttt{BEremoval}(X_{tr}, c, y_{tr})$
4: $idx_{bestGenes} \leftarrow \texttt{ttest}(y_{tr}, \hat{X}_{tr})$
5: $X_{class} \leftarrow \hat{X}_{tr}[idx_{bestGenes}, :]$
6: $model_{class} \leftarrow \texttt{classifier}(X_{class}, y_{tr})$
7: $\hat{X}_{te} \leftarrow \texttt{BEremoval}(X_{te}, param_{BE})$
8: $\hat{y}_{te} \leftarrow \texttt{prediction}(model_{class}, \hat{X}_{te}[idx_{bestGenes}, :])$

We can see on Figure 2 that in some cases such as $RatioA$ or $rgCCA_{sq}$, some methods appear to worsen the BCR compared to the reference case without any batch effect removal. Datasets GSE5327_{96} and GSE32641_{887} have really bad results (BCR ≤ 0.5), probably due to the fact that their class frequencies (100% of ER-) are really far from the training ones (majority of ER+). From now on, we will exclude those two datasets from the testing set. All other datasets have a quite similar behavior, except for GSE6532_{96} which leads to bad results for location-scale methods.

Detailed results for dataset GSE6532_{96} are shown on Figure 3, where we can see that applying a specific method can really decrease or increase the performances. Any location-scale method worsens the results while FA and SVD improve it significantly. The bad behavior of location-scale methods can probably be explained by the two clusters initially present in the dataset and that cannot be removed by a simple location-scale method (see Figure 1). On the contrary, matrix factorization methods, in view of their unsupervised foundations (namely, line 1 of Table I is unsupervised), are able to capture trends within batches.

2) Influence of classifier: Similarly, the effects of the different batch effect removal methods and classifiers on the testing sets were compared on Figure 4. We can see on Figure 4 that all classifiers appear to have a similar behavior for the majority of methods. Some methods such as $RatioA$, SVA or $rgCCA_{sq}$ worsen the BCR while FA, SVD and $stICA_{0.25,0.5}$ lead to the best results, with a slightly higher mean and a smaller variance than $None$. Note that while using class information, SVA performances are among the worse, probably due to the fact that the method does not use the batch information.

3) Influence of technology used: As initially the datasets tend to cluster by the technology used (Affymetrix or Agilent), we investigated more in depth the influence of this factor in the training and testing sets on Figure 5. For this purpose two more classification models were trained, one on the 3

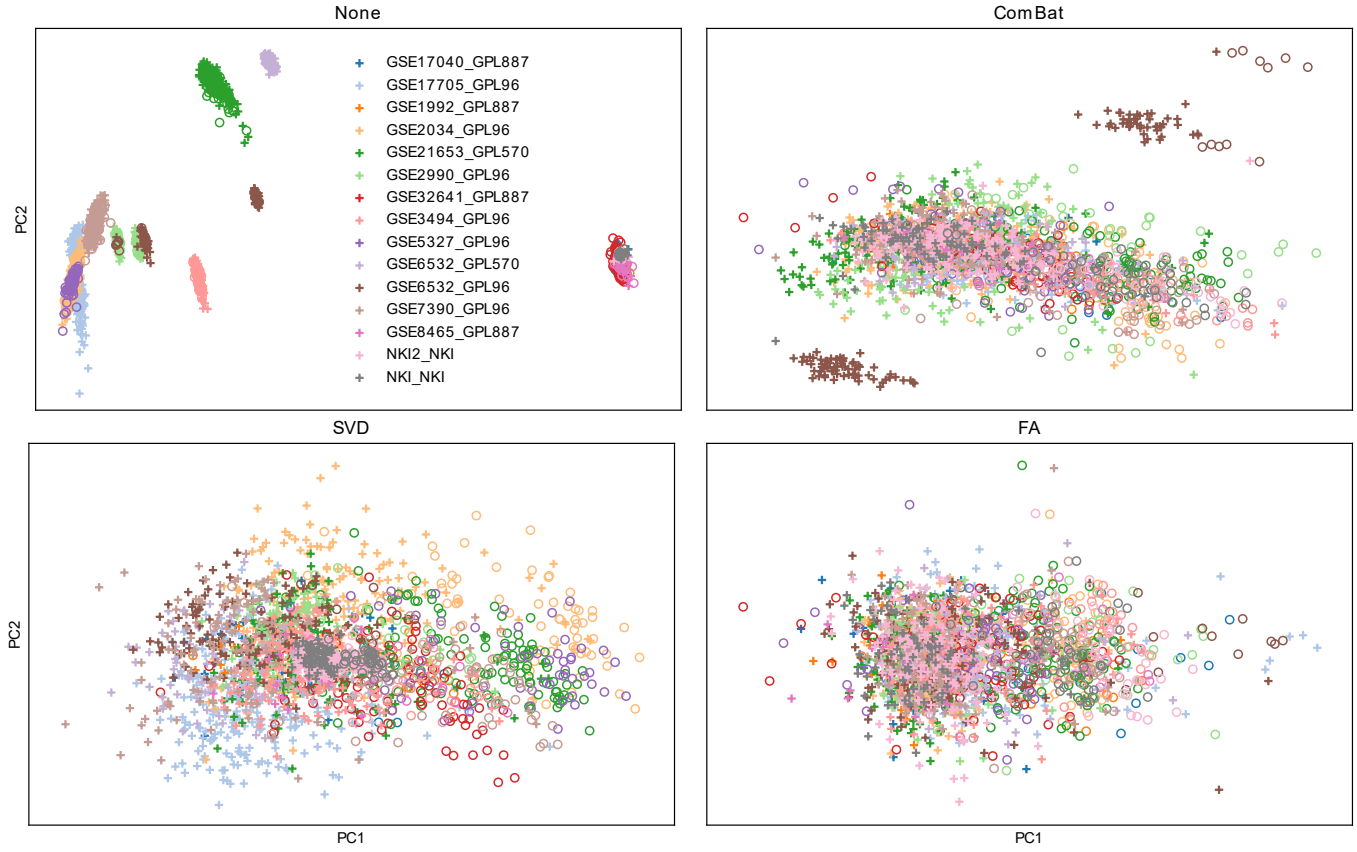

Fig. 1. Global evaluation of batch effects: plots of the first two principal components before and after batch effect removal by ComBat, FA and SVD. Colors represent the batch membership, $o/+$ corresponds to ER-/ER+ status.

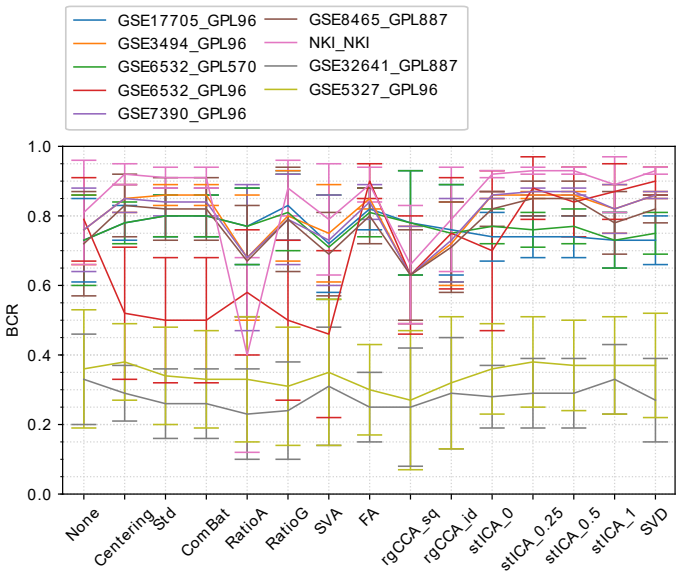

Fig. 2. Influence of the batch effect removal methods on BCR for the different testing datasets. Mean on all training datasets, on all classifiers and on all numbers of selected features.

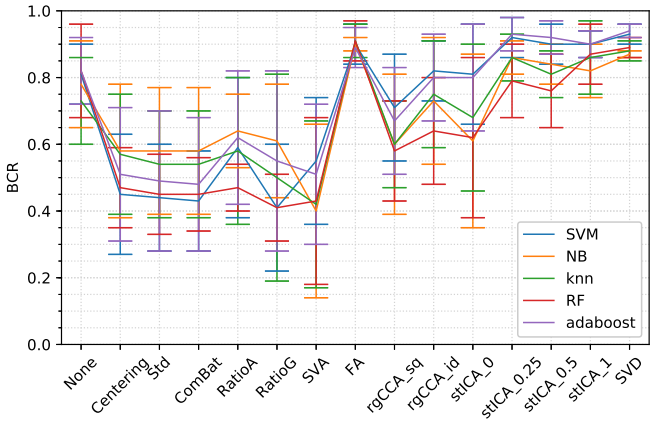

Fig. 3. Influence of the batch effect removal methods and classifiers on BCR for $GSE6532_{96}$

Agilent training datasets only and one on the 3 Affymetrix ones. Testing set was also separated in two, depending on the technology used. Difference between both technologies are limitedbut matrix factorization methods seem to be less sensitive to the technology used.

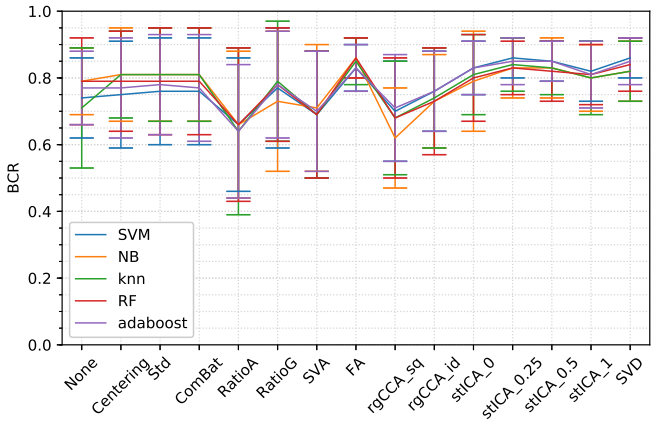

Fig. 4. Influence of the batch effect removal methods and classifiers on BCR. Mean on test datasets and on numbers of selected features.

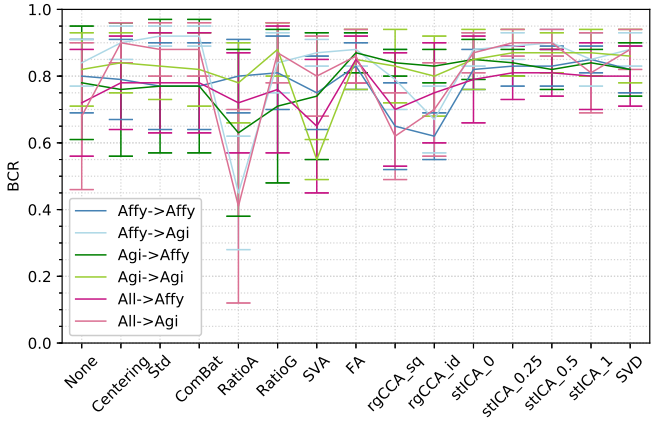

Fig. 5. Influence of the batch effect removal methods and technology used on BCR. $X \rightarrow Y$ means that the model was trained on datasets using technology X, and tested on technology Y. 'All' corresponds to the means on the C_6^4 models precedently trained.

4) Influence of training set: Figure 6 shows the average performances depending on the training set used. Whereas the results depend strongly on the training test when no method is used, applying any batch effect removal method increases the stability of results. The best methods regarding stability are still the same: FA, SVD, $stICA_{0.25,0.5}$. $ComBat$, Std and $Centering$ are slightly below due to $GSE6532_{96}$.

5) Influence of class frequencies in the training set: As seen previously, the models used have real difficulties to make correct predictions for datasets with only ER- samples. To investigate the effect of imbalanced repartition of classes in the training set, we trained our models on training datasets randomly subsampled to increase the proportion of ER- samples (described in Table IV). Performances obtained on those training datasets are shown on Figure 7. The two worse test datasets are the two with only ER+ samples. All datasets with a majority of ER+ samples have degraded BCR while performances of those with ER- only have improved compared to Figure 2. General behavior of the different methods is similar to the previous observations, except for FA which is now among the worse methods.

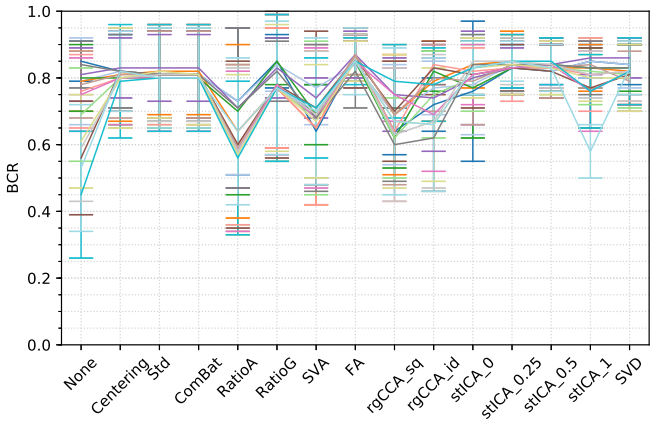

Fig. 6. Influence of batch effect removal methods and training sets on BCR, averaged on all test datasets and all classifiers. Each curve corresponds to a different training set.

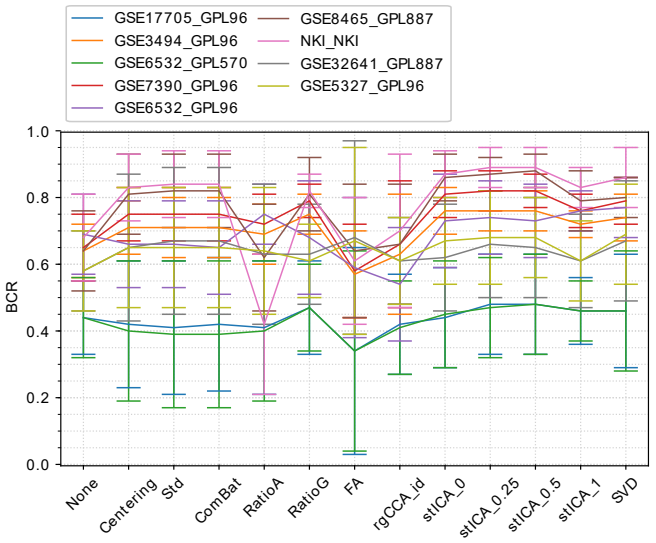

Fig. 7. Influence of the batch effect removal methods on BCR for the different testing datasets when training set is modified to have a higher proportion of ER- samples.

IV. Conclusion

In the context of merging gene expression datasets, we have investigated two types of batch effect removal approaches: location-scale and matrix factorization based methods. We have compared those methods used as preprocessing tools for various basic classifiers. To comply with real-life scenario, we first applied the batch effect removal method, selected the features and trained the classifiers on a subset of datasets. Once all the models were trained based on the training set, we then applied on a separated test set the batch effect removal method and the classifier using the previously fixed models.

The first observation is that in a specific case (dataset $GSE6532_{96}$), applying location-scale methods can worsen the

TABLE IV
SUMMARY OF BATCHES SUBSAMPLED TO MODIFY THE CLASS FREQUENCIES.

	Batch	Technology	# ER-	# ER+
Train	$GSE2034_{96}$	Affymetrix	77	19
	$GSE21653_{570}$	Affymetrix	113	28
	$GSE2990_{96}$	Affymetrix	21	85
	$GSE17040_{887}$	Agilent	11	44
	$GSE1992_{887}$	Agilent	6	23
	NKI2	Agilent	43	11

classification performances. This may be due to the presence of a batch effect within the dataset. On the contrary, matrix factorization based methods allows to significantly improve the performances.

In most cases we examined, applying a batch effect removal method did not increase a lot the performances, some even nearly systematically decreasing it ($RatioA$, SVA, and $rgCCA$). However even those methods appear to decrease the influence of the training set used (see Figure 6), which is essential in the reproducibility of results. The best methods in our tests are FA, SVD, and $stICA_{0.25,0.5}$. FA being among the best methods is not surprising as it uses the class information in the training set, whereas the other methods do not use at all the class information. In other words, FA is a classification-task-aware method, whereas the other batch effect removal methods are more general purpose. More unexpected is that SVD and some of the ICA based methods, that unlike FA uses only batch information, behave as well as FA; see Figure 4. Note that if we do not consider dataset GSE6532, location-scale methods $ComBat$, Std and $Centering$ behave as well as the previous ones.

Class frequency in the training set influences also the results, and extra caution should be taken in presence of imbalanced repartition of the classes among the datasets, especially with supervised methods such as FA.

To conclude, we have shown that matrix factorization based methods behave at least as well as more common location-scale methods, and even better when all the batch information is not specifically known. We only tested a few factorizations, but different approaches such as non-negative matrix factorization or other versions of $rgCCA$ should be investigated. Such methods are often used in multi-omics data integration, but could also be adapted in the case of batch effect removal.

REFERENCES

[1] J. T. Leek and J. D. Storey, "Capturing heterogeneity in gene expression studies by surrogate variable analysis," *PLoS Genet*, vol. 3, no. 9, p. e161, 09 2007.

[2] J. T. Leek *et al.*, "Tackling the widespread and critical impact of batch effects in high-throughput data." *Nat Rev Genet*, vol. 11, no. 10, pp. 733–739, Oct. 2010.

[3] A. E. Teschendorff, J. Zhuang, and M. Widschwendter, "Independent surrogate variable analysis to deconvolve confounding factors in large-scale microarray profiling studies." *Bioinformatics*, vol. 27, no. 11, pp. 1496–1505, 2011.

[4] A. A. Shabalin, H. Tjelmeland, C. Fan, C. M. Perou, and A. B. Nobel, "Merging two gene-expression studies via cross-platform normalization," *Bioinformatics*, vol. 24, no. 9, pp. 1154–1160, 2008.

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

[6] J. Luo *et al.*, "A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data," *The pharmacogenomics journal*, vol. 10, no. 4, pp. 278–291, 2010.

[7] W. E. Johnson, 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.

[8] P. Warnat, R. Eils, and B. Brors, "Cross-platform analysis of cancer microarray data improves gene expression based classification of phenotypes," *BMC bioinformatics*, vol. 6, no. 1, p. 1, 2005.

[9] 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.

[10] H. Yasrebi, "Comparative study of joint analysis of microarray gene expression data in survival prediction and risk assessment of breast cancer patients," *Briefings in bioinformatics*, vol. 17, no. 5, pp. 771–785, 2015.

[11] O. Alter, P. O. Brown, and D. Botstein, "Singular value decomposition for genome-wide expression data processing and modeling," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 97, no. 18, pp. pp. 10 101–10 106, 2000.

[12] E. Renard, S. Branders, and P.-A. Absil, "Independent component analysis to remove batch effects from merged microarray datasets," in *Algorithms and Bioinformatics - 16th International Workshop, WABI 2016, Aarhus, Denmark, August 22-24, 2016. Proceedings*, ser. Lecture Notes in Bioinformatics. Springer, 2016, vol. 9838, pp. 281–292.

[13] F. Rohart, A. Eslami, N. Matigian, S. Bougeard, and K.-A. Le Cao, "Mint: A multivariate integrative method to identify reproducible molecular signatures across independent experiments and platforms," *BMC bioinformatics*, vol. 18, no. 1, p. 128, 2017.

[14] C. Lazar *et al.*, "Batch effect removal methods for microarray gene expression data integration: a survey," *Briefings in bioinformatics*, vol. 14, no. 4, pp. 469–490, 2013.

[15] J. Taminau, C. Lazar, S. Meganck, and A. Nowé, "Comparison of merging and meta-analysis as alternative approaches for integrative gene expression analysis," *ISRN bioinformatics*, vol. 2014, 2014.

[16] J. Rudy and F. Valafar, "Empirical comparison of cross-platform normalization methods for gene expression data," *BMC bioinformatics*, vol. 12, no. 1, p. 467, 2011.

[17] C. Chen *et al.*, "Removing batch effects in analysis of expression microarray data: an evaluation of six batch adjustment methods," *PloS one*, vol. 6, no. 2, p. e17238, 2011.

[18] R. Hornung, D. Causeur, C. Bernau, and A.-L. Boulesteix, "Improving cross-study prediction through addon batch effect adjustment or addon normalization - tq report," *Bioinformatics*, vol. 33, no. 3, pp. 397–404, 2017.

[19] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, no. 3/4, pp. 321–377, 1936.

[20] A. Tenenhaus and M. Tenenhaus, "Regularized generalized canonical correlation analysis," *Psychometrika*, vol. 76, no. 2, pp. 257–284, 2011.

[21] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[22] E. Renard, A. E. Teschendorff, and P. Absil, "Capturing confounding sources of variation in dna methylation data by spatiotemporal independent component analysis," in *22nd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2014)*, 2014.

[23] R. Hornung, A.-L. Boulesteix, and D. Causeur, "Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment," *BMC bioinformatics*, vol. 17, no. 1, p. 27, 2016.