LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

INSTITUT FÜR STATISTIK

Roman Hornung, Anne-Laure Boulesteix, David Causeur

Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment

Technical Report Number 184, 2015
Department of Statistics
University of Munich

http://www.stat.uni-muenchen.de

Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment

Roman Hornung[1*] Anne-Laure Boulesteix[1] David Causeur[2]

September 22, 2015

[1] Department of Medical Informatics, Biometry and Epidemiology, University of Munich, Marchioninistr. 15, D-81377, Munich, Germany

[2] Agrocampus Ouest, Applied Mathematics Department, 35000 Rennes, France

Abstract

In the context of high-throughput molecular data analysis it is common that the observations included in a dataset form distinct groups; for example, measured at different times, under different conditions or even in different labs. These groups are generally denoted as batches. Systematic differences between these batches not attributable to the biological signal of interest are denoted as batch effects. If ignored when conducting analyses on the combined data, batch effects can lead to distortions in the results. In this paper we present FAbatch, a general, model-based method for correcting for such batch effects in the case of an analysis involving a binary target variable. It is a combination of two commonly used approaches: location-and-scale adjustment and data cleaning by adjustment for distortions due to latent factors. We compare FAbatch extensively to the most commonly applied competitors on the basis of several performance metrics. FAbatch can also be used in the context of prediction modelling to eliminate batch effects from new test data. This important application is illustrated in a real data application. We implemented FAbatch and various other functionalities in the R package `bapred` available online from CRAN. FAbatch is seen to be competitive in many cases and above average in others. In our analyses, the only cases where it failed to adequately preserve the biological signal were when there were extremely outlying batches and when the batch effects were very weak compared to the biological signal. As seen in this paper batch effect structures found in real datasets are diverse. Current batch effect adjustment methods are often either too simplistic or make restrictive assumptions, which

*Corresponding author. Email: hornung@ibe.med.uni-muenchen.de.

can be violated in real datasets. Due to the generality of its underlying model and its ability to perform well FAbatch represents a reliable tool for batch effect adjustment for most situations found in practice.

Background

In practical data analysis, the observations included in a dataset sometimes form distinct groups—denoted as "batches"; for example, measured at different times, under different conditions, by different persons or even in different labs. Such batch data is common in the context of high-throughput molecular data analysis, where experimental conditions typically have a high impact on the measurements and only few patients are considered at a time. Taking a more general point of view, different batches may also represent different studies concerned with the same biological question of interest. Independently of the particular scenario, in this paper all systematic differences between batches of data not attributable to the biological signal of interest are denoted as batch effects. If ignored when conducting analyses on the combined data, batch effects can lead to distorted and less precise results.

It is clear that batch effects are more severe when the sources from which the individual batches originate are more disparate. Batch effects—in our definition—may also include systematic differences between batches due to biological differences of the respective populations unrelated to the biological signal of interest. This conception of batch effects is related to an assumption made on the distribution of the data of recruited patients in randomized controlled clinical trials (see, e.g., [16]). This assumption is that the distribution of the (metric) outcome variable may be different for the actual recruited patients than for the patients eligible for the trial, i.e. there may be biological differences, with one important restriction: the difference between the means in treatment and control group must be the same for recruited and eligible patients. Here, the population of recruited patients and the population of eligible patients can be perceived as two batches (ignoring that the former population is a—very small—subset of the latter) and the difference between the means of the treatment and control group would correspond to the biological signal.

Various methods have been developed to correct for batch effects. See for example [11] and [15] for a general overview and for an overview of methods suitable in applications involving prediction, respectively. Two of the most commonly used methods are ComBat [9], a location-and-scale batch effect adjustment method and SVA [13, 17], a non-parametric method, in which the batch effects are assumed to be induced by latent factors. Even though the assumed form of batch effects underlying a location-and-scale adjustment as done by ComBat is rather simple, this method has been

observed to greatly reduce batch effects [4]. However, a location-and-scale model is often too simplistic to account for more complicated batch effects. SVA is, unlike ComBat, concerned with situations where it is unknown which observations belong to which batches. This method aims at removing inhomogeneities within the dataset that also distort its correlation structure. These inhomogeneities are assumed to be caused by latent factors. When the batch variable is known, it is natural to take this important information into account when correcting for batch effects. Also, it is reasonable here to make use of the data-cleaning ability of the latent factor-adjustment by applying it within batches. This has the effect of reducing such inhomogeneities within batches, which are unrelated to the biological signal of interest. By doing so it can be expected that the homogeneity of the data is further increased across batches as well.

In this paper we suggest a method, denoted as "FAbatch" in the following, where "FA" stands for "**F**actor **A**djustment". The method combines the location-and-scale adjustment (as performed by ComBat) with data cleaning by latent factor adjustment (as performed by SVA). Care has to be taken in the latent factor estimation in the context of data-cleaning. Inhomogeneities within the dataset are naturally not only induced by sources of unwanted noise but also by the biological signal of interest. If one would not take this interference between batch effects and signal into account, removing the corresponding estimated latent factor loadings would lead to removing a large part of the biological signal of interest. An obvious, yet problematic way, of protecting the signal of interest would be to remove it temporarily before estimating the latent factors by regressing each of the variables in the dataset on the variable representing the biological signal. However, this can lead to an artificially increased signal, as outlined in the section 'FAbatch `(fabatch)`'. As a solution for the case of a binary variable representing the biological signal, in our method we fit preliminary L_2-penalized logistic regression models and use them to predict the probabilities of the individual observations to belong to the first and the second class, respectively. These predicted probabilities are then used in place of the actual values of the binary variable when protecting the signal of interest during latent factor estimation. See the section 'FAbatch `(fabatch)`' for details. In its current form our method is thus only applicable when the signal variable is binary, but extensions to other types of variables are possible, see the section 'Discussion'.

Very importantly, our method allows for the elimination of batch effects *a posteriori* in independent observations from different batches with unknown signal variable, most commonly for the purpose of performing prediction. Note that a posteriori adjusting independent batches is not restricted to our method, but a general concept referred to as "addon batch effect adjustment" in the following. Here, first batch effect adjustment is conducted based on the available original dataset. Some methods require that the val-

ues of the target variable are known in this dataset. Subsequently, batch effect adjustment for independent batches is performed. To facilitate this, it is required that several observations from each batch are available simultaneously ("frozen SVA" is an exception here, see the section 'Addon adjustment of independent batches'). This second phase does not affect the data prepared in the first phase. See the section 'Addon adjustment of independent batches' for details. In the context of prediction, it is thus possible to eliminate batch effects from independent data before applying a prediction rule that was previously fitted on the original data. This feature of our method is very important in practice, since training data and validation data often consist of different batches. We refer to such scenarios as cross-batch prediction in the rest of this paper.

As an illustration, Figure 1 shows plots of the first two principal components obtained by Principal Component Analysis (PCA) on a raw dataset (upper-left) and after running the three different batch effect adjustment methods described above, respectively. The dataset, composed of two batches, contains the gene expressions of 20 alcoholics and 19 healthy controls. It is downloadable from ArrayExpress [10], accession number: E-GEOD-44456. After ComBat adjustment, the centers of gravity of the first principal components separated into the two batches become very similar (upper-right panel). However, the shapes of the point clouds corresponding to the two batches do not change substantially in comparison to the results obtained on the raw data (upper-left panel) and the two clouds do not fully overlap. After SVA adjustment—as with ComBat—the two batch centers are also similar (lower-left panel). The forms of the point clouds change more strongly compared to ComBat. Nevertheless, there are still regions in the plots with suboptimal overlap between the two clouds. The two batch centers are not distinguishable in the plot showing the result obtained after applying our method (lower-right panel). Moreover, the overlap between the two clouds is very high. This illustrative example suggests that the adjustment for batch effects can be improved by combining location-scale-adjustment with data-cleaning by factor adjustment.

The structure of this paper is as follows: In the section 'Methods' we first give an overview of the ComBat-method and the SVA-method and then introduce our new approach, demonstrating that it can be seen as an extension of ComBat by batchwise adjustment for latent factor influences similar to the application of SVA within batches. Furthermore we describe already existing batch effect adjustment methods. Subsequently we explain how to adjust for batch effects a posteriori in independent observations from different batches, which is important in cross-batch prediction, as mentioned above. We outline the corresponding procedure in general and for the specific batch effect adjustment methods considered in this paper. In the following subsection we present the design of an extensive comparison study based on simulations and real data applications. In this study our method is compared with

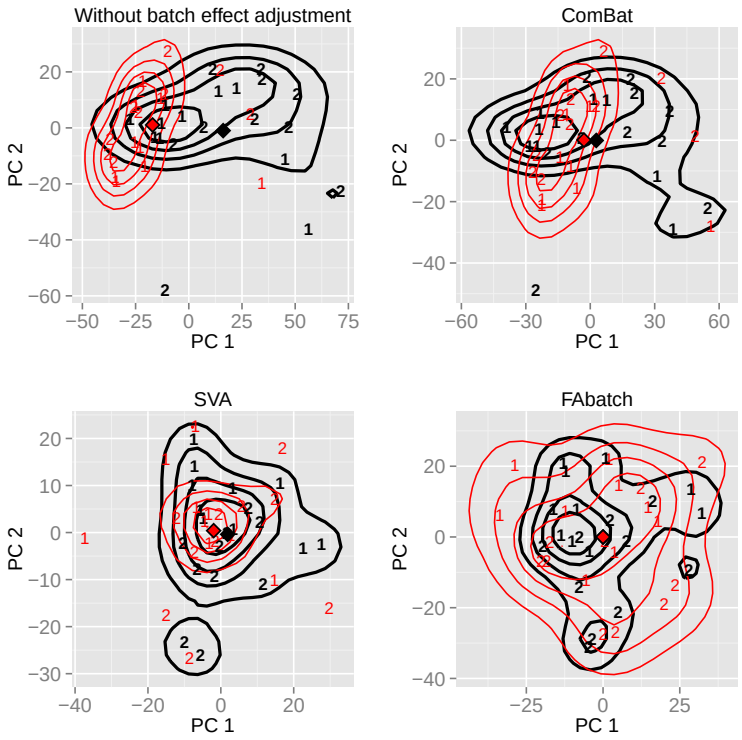

Figure 1 Visualization of batch effect adjustment. First two principal components out of PCA performed on the covariate matrix of a microarray dataset studying alcoholism: raw, after batch effect adjustment according to ComBat, SVA using three factors and FAbatch using three factors. The first batch is depicted in bold and the numbers distinguish the two classes "alcoholic" (2) vs. "healthy control" (1). The contour lines represent batch-wise two-dimensional kernel estimates and the diamonds represent the batch-wise centers of gravities of the points.

commonly used competitors with respect to diverse metrics measuring the effectiveness of batch effect adjustment [11, 12]. The considered methods are: FAbatch, ComBat, SVA, mean-centering, standardization, ratio-A and ratio-G, see the section 'Comparison of FAbatch with existing methods' for more details. The results of this study are described in the section 'Results', where we also present a real data example demonstrating the use of batch effect adjustment methods in cross-batch prediction. The section 'Discussion' mostly reviews the models behind FAbatch and other approaches, and the section 'Conclusions' summarizes important conclusions from the paper.

Methods

ComBat (`comb`)

This method assumes the following model for the observed data x_{ijg}:

$$x_{ijg} = \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \gamma_{jg} + \delta_{jg}\epsilon_{ijg}, \quad \epsilon_{ijg} \sim N(0, \sigma_g^2). \tag{1}$$

Here i is the index for the observation, j the index for the batch and g the index for the variable. The term $\boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g$ parametrizes the effect of experimental conditions or, in general, any factors of interest $\boldsymbol{a}_{ij}$ on the measurements of variable g. In this paper, $\boldsymbol{a}_{ij} \in \{0, 1\}$ represents the binary variable of interest, which is assumed to be common to all considered datasets. Note that this restriction to binary target variables, which is not necessary in general for the application of ComBat, is required for the application of our method (see the section 'FAbatch (`fabatch`)').

The term γ_{jg} corresponds to the mean shift in location of variable g in the j-th batch compared to the unobserved—hypothetical—data x^*_{ijg} unaffected by batch effects. The term δ_{jg} corresponds to the scale shift in the j-th batch.

The unobserved counterpart x^*_{ijg} of x_{ijg} not affected by batch effects is assumed to be

$$x^*_{ijg} = \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \epsilon_{ijg}, \quad \epsilon_{ijg} \sim N(0, \sigma_g^2). \tag{2}$$

The goal of batch effect correction via ComBat is to estimate these unobserved x^*_{ijg}-values. The following transformation of the observed x_{ijg}-values would provide the true x^*_{ijg}-values:

$$\sqrt{\mathrm{Var}(x^*_{ijg})}\left(\frac{x_{ijg} - \mathrm{E}(x_{ijg})}{\sqrt{\mathrm{Var}(x_{ijg})}}\right) + \mathrm{E}(x^*_{ijg}) \tag{3}$$

$$= \sigma_g\left(\frac{x_{ijg} - (\alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \gamma_{jg})}{\delta_{jg}\sigma_g}\right) + \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g \tag{4}$$

$$= \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \epsilon_{ijg} = x^*_{ijg}. \tag{5}$$

In practice, however, the parameters involved in Eq. (4) are not known and have to be estimated. In particular, γ_{jg}/σ_g and δ_{jg} are estimated using empirical Bayes to obtain more robust results. See [9] for details on the estimation procedure. Note that in the analyses performed in this paper we do not include the term $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ in the adjustment. The first reason for this is that in the section 'Application in cross-batch prediction' we study cross-batch prediction. Here the class values $\boldsymbol{a}_{ij}$ are not known in the test data. The second reason is that using the class values $\boldsymbol{a}_{ij}$ together with the estimates of $\boldsymbol{\beta}_g$ may lead to an artificially increased class signal, because the estimates of $\boldsymbol{\beta}_g$ depend on the class values $\boldsymbol{a}_{ij}$. This kind of mechanism will be discussed in detail, but in slightly other contexts, in the sections 'Using estimated probabilities instead of actual classes' and 'Application in cross-batch prediction'.

SVA (`sva`)

The model for the observed data is given by:

$$x_{ijg} = \alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \sum_{l=1}^{m} b_{gl}Z_{ijl} + \epsilon_{ijg}, \tag{6}$$

$$\mathrm{Var}(\epsilon_{ijg}) = \sigma_g^2. \tag{7}$$

Here α_g and $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ are as in the section 'ComBat (`comb`)', $Z_{ij1}, \ldots, Z_{ijm}$ are random latent factors and $b_{g1}, \ldots, b_{gm}$ are variable-specific regression coefficients, denoted as factor loadings.

The unobserved, batch-free data is correspondingly:

$$x_{ijg}^* = \alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \epsilon_{ijg}, \quad \mathrm{Var}(\epsilon_{ijg}) = \sigma_g^2. \tag{8}$$

Note again that in the SVA-model the batch membership is assumed to be unknown. For judging the appropriateness of the SVA algorithm it is important to specify the model underlying SVA as precisely as possible. Out of the following two facts it can be followed that the distribution of the latent factors can be different for each observation—in the extreme case. Firstly, the assumed form of the batch-free data in Eq. (8) implies that the distortions between the batches are induced fully by the latent factors. Secondly, each observation may come from a different batch with own mean-, covariance- and correlation-structure.

The SVA batch effect adjustment is performed by substracting $\sum_{l=1}^{m} b_{gl}Z_{ijl}$ from x_{ijg}:

$$x_{ijg} - \sum_{l=1}^{m} b_{gl}Z_{ijl} = \alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \epsilon_{ijg} = x_{ijg}^*. \tag{9}$$

The latent factors are estimated as the first m right singular vectors from a singular value decomposition (SVD). In the section 'Background' we stressed

that inhomogeneities in datasets are not only due to batch effects, but also due to the biological signal of interest, i.e. the term $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ in Eq. (6) and (8). Therefore, we noted that the biological signal of interest has to be protected during factor estimation in FAbatch. In SVA, to protect the biological signal, before performing the SVD on the transposed covariate matrix, the variable values are weighted by the estimated probabilities that the corresponding variables are associated with unmeasured confounders, but not with the binary variable representing the biological signal. The factor loadings are estimated by linear models. The "frozen SVA" procedure [17] is an extension of SVA [13] developed for the purpose of prediction, see the section 'Addon adjustment of independent batches'.

FAbatch (`fabatch`)

Model

We assume the following model for the observed data:

$$\begin{aligned} x_{ijg} &= \alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \gamma_{jg} + \sum_{l=1}^{m_j} b_{jgl}Z_{ijl} + \delta_{jg}\epsilon_{ijg}, \\ Z_{ij1},\ldots,Z_{ijm_j} &\sim N(0,1), \quad \epsilon_{ijg} \sim N(0,\sigma_g^2), \end{aligned} \tag{10}$$

where α_g, $\boldsymbol{a}_{ij}^T$, $\boldsymbol{\beta}_g$, γ_{jg} and δ_{jg} are defined as in ComBat. As in the SVA model, Z_{ijl} are random latent factors. Note that here the factor loadings b_{jgl} are batch-specific.

The unobserved data x_{ijg}^* not affected by batch effects is assumed to have the form

$$x_{ijg}^* = \alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \epsilon_{ijg}, \quad \epsilon_{ijg} \sim N(0,\sigma_g^2). \tag{11}$$

Note that the model of the observed data in (10) is obviously an extension of the model underlying ComBat by the contribution $\sum_{l=1}^{m_j} b_{jgl}Z_{ijl}$ of batch-specific latent factors. While both SVA and FAbatch involve latent factors, these and their roles are quite different in the two models. In the SVA model, each observation may represent a batch on its own and the batch effects are assumed to be fully induced by the latent factors. On the contrary, in the FAbatch model we assume that the batches are known and that the systematic differences between the batches are in part—but not in total—explainable by location-scale-shifts between the batches. A simple location-and-scale-shift may not be sufficient in situations with complicated batch effects, as illustrated in the example in the section 'Background'. The latent factors in the FAbatch model allow us to account for batch effects within the individual batches, which are not rendered by the location-and-scale shifts performed by ComBat. In contrast to the SVA model, in our model the distribution of the latent factors is independent of the individual observation.

However, since the loadings of the latent factors are batch-specific, the latter induce batch effects in our model as well. More precisely, they lead to varying correlation structures in the batches.

Using estimated probabilities instead of actual classes

As already noted in the section 'Background', a further peculiarity of our method is that we do not use the actual classes when protecting the biological signal of interest in the estimation algorithm. Instead, we estimate the probabilities of the observations to belong to either class and use these in place of the actual classes, see the next paragraph and the next subsection for details.

This procedure has two major advantages. Firstly, it makes the batch effect correction method applicable to prediction problems involving new test observations with unknown classes. Secondly, using the actual classes might lead to an artificial increase of separation between the two classes in the dataset. This is because, as will be seen in the estimation algorithm, it is necessary to use the estimated, instead of the true, but unknown, class-specific means when centering the data before factor estimation. Due to sampling variance, these estimated class-specific means often lie further away from each other than the true means, in particular for variables for which the true means lie close to each other. Subtracting the estimated factors' influences leads to a reduction of the variance. Now, if centering the variable values within the classes before factor estimation, removing the estimated factor influences would lead to a reduction of the variance around the respective estimated class-specific means. In those—frequently occurring—cases, in which the estimated class-specific means lie further from each other than the corresponding true means, this would lead to an artificial increase of the discriminatory power of the corresponding variable in the adjusted dataset.

All analyses which are concerned with the discriminatory power of the covariate variables with respect to the target variable would be biased if performed on data adjusted in this way. More precisely, the discriminatory power would be overestimated. This mechanism is conceptually similar to the over-fitting of prediction models on the data they were obtained on. SVA suffers from a very similar kind of bias, also related to using the class information in protecting the biological signal. See the section 'Application in cross-batch prediction' for a detailed description of this phenomenon and the results of a small simulation study performed to assess the impact of this bias on data analysis in practice.

In the FAbatch-method this problem is avoided by using estimates of the probabilities of the observations to belong to either class in place of the actual class values. These probabilities are estimated using models fitted from data other than the corresponding observations. This procedure attenuates the artificial increase of the class signal described above. The idea underly-

ing this procedure is to center x_{ijg} before factor estimation by subtracting the term

$$\begin{aligned}&\mathbb{E}(\alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \gamma_{jg}|x_{ij1,},\ldots,x_{ijp}) = \\ &\quad \Pr(y_{ij}=1|x_{ij1,},\ldots,x_{ijp})\ (\alpha_g+\gamma_{jg}) + \\ &\quad \Pr(y_{ij}=2|x_{ij1,},\ldots,x_{ijp})\ (\alpha_g+\boldsymbol{\beta}_g+\gamma_{jg}).\end{aligned} \tag{12}$$

Note that we perform this adjustment slightly differently in the FAbatch-estimation algorithm. See the next section for details.

Estimation

In the following we detail the estimation procedure of FAbatch:

1. Standardize the values x_{ijg} per batch:

$$x_{ijg,S} := \frac{x_{ijg} - \widehat{\mu_{jg}}}{\sqrt{\widehat{\sigma_{jg}^2}}}, \tag{13}$$

   where $\widehat{\mu_{jg}} = (1/n_j)\sum_i x_{ijg}$ and $\widehat{\sigma_{jg}^2} = [1/(n_j-1)]\sum_i (x_{ijg} - \widehat{\mu_{jg}})^2$. Here, the number of observations in batch j is denoted as n_j.

2. Using L_2-penalized logistic regression, for each observation estimate the probability to belong to class 2:

$$\widehat{\pi_{ij}} := \widehat{\Pr}(y_{ij}=2|x_{ij1,S},\ldots,x_{ijp,S}). \tag{14}$$

   Here, we employ the following cross-validation related procedure. For batch $j \in \{1,\ldots,K\}$: 1) Fit a L_2-penalized logistic regression model using all observations except those in batch j; 2) Use the model fitted in step 1) to predict the probabilities π_{ij} of the observations from batch j. By using different observations for fitting the models than for predicting the probabilities we avoid overfitting in the sense of the problems occurring when the actual classes are used as described in the previous subsection. The reason why we perform cross-batch prediction for estimating the probabilities here instead of ordinary cross-validation is that we expect the resulting batch adjusted data to be more suitable for the application in cross-batch prediction (see the section 'Addon adjustment of independent batches'). Here, for estimating the probabilities in the test batch we have to use a prediction model fitted on other batches. If the probabilities in the training data were estimated via ordinary cross-validation they would be more optimistic—i.e. closer to zero and one, respectively—than those in the test data. This is because in ordinary cross-validation it can occur that observations from the same batch are in training and test data. By

doing cross-batch prediction for the estimation of the π_{ij} we mimic the situation encountered in cross-batch prediction applications. The only exception where we perform ordinary cross-validation for estimating the π_{ij} is when the data come from only one batch (this occurs in the context of cross-batch prediction, when the training data consist of one batch).

The shrinkage intensity tuning parameter of the L_2-penalized logistic regression model is optimized with the aid of cross-validation [8]. For computational efficiency this optimization is not repeated in each iteration of the cross-batch prediction. Instead, it is performed beforehand on the complete dataset. The overoptimism resulting from this procedure compared to that of the gold-standard technique "nested cross-batch prediction" can be assumed to be negligible in the considered context.

3. Calculate the class adjusted values $x_{ijg,S,CA}$, which should contain considerably less class signal than $x_{ijg,S}$:

$$x_{ijg,S,CA} := x_{ijg,S} - (1 - \widehat{\pi_{ij}})\widehat{\mu_{g,S}}^{(1)} - \widehat{\pi_{ij}}\widehat{\mu_{g,S}}^{(2)}, \tag{15}$$

where $\widehat{\mu_{g,S}}^{(c)} = (1/\#L_c)\sum_{\{i^*,j^*\}\in L_c} x_{i^*j^*g,S}$ with $L_c = \{\{i,j\} : y_{ij} = c, i \in \{1,\ldots,n_j\}, j \in \{1,\ldots,J\}\}$ and $c \in \{1,2\}$.

4. Using $x_{ijg,S,CA}$, estimate the latent factors $Z^*_{ijm_j}$ and their loadings $b^*_{jgm_j}$ by an EM-algorithm presented in [18], again considered by Friguet et al. [5] in a specific context for microarray data. For the estimation of the number of factors see [5].

5. Subsequently the estimated factor contributions are removed:

$$x_{ijg,S,FA} := x_{ijg,S} - \widehat{b^*_{jg1}}\widehat{Z^*_{ij1}} - \cdots - \widehat{b^*_{jgm_j}}\widehat{Z^*_{ijm_j}}, \tag{16}$$

where $\widehat{b^*_{jg1}}, \ldots, \widehat{b^*_{jgm_j}}$ are the estimated, batch-specific factor loadings and $\widehat{Z^*_{ij1}}, \ldots, \widehat{Z^*_{ijm_j}}$ are the estimated latent factors. Note that only the factor contributions as a whole are identifiable, not the individual factors and their coefficients.

6. Finally, in each batch the $x_{ijg,S,FA}$-values are transformed to have the global means and pooled variances estimated before batch effect adjustment:

$$\widehat{x^*_{ijg}} = \left(\frac{x_{ijg,S,FA} - \widehat{\mu_{g,S,FA}}}{\sqrt{\widehat{\sigma^2_{g,S,FA}}}}\right)\sqrt{\widehat{\sigma^2_g}} + \widehat{\mu_g}, \tag{17}$$

where
$$\widehat{\mu_{g,S,FA}} = \Big(1/\sum_j n_j\Big)\sum_j\sum_i x_{ijg,S,FA},$$
$$\widehat{\sigma^2_{g,S,FA}} = \Big[1/\Big(\sum_j n_j - 1\Big)\Big] \sum_j\sum_i (x_{ijg,S,FA} - \widehat{\mu_{g,S,FA}})^2,$$
$$\widehat{\mu_g} = \Big(1/\sum_j n_j\Big)\sum_j\sum_i x_{ijg}$$
and
$$\widehat{\sigma^2_g} = \Big[1/\Big(\sum_j n_j - 1\Big)\Big]\sum_j\sum_i (x_{ijg} - \widehat{\mu_g})^2.$$

Note that by forcing the empirical variances in the batches to be equal to the pooled variances estimated before batch effect adjustment we overestimate the residual variances σ_g^2 in (10). This is because we do not take into account that the variance is reduced by the adjustment for latent factors. However, unbiasedly estimating σ_g^2 appears difficult due to the scaling before estimation of the latent factor contributions.

Further batch effect adjustment methods considered in comparison studies

In addition to FAbatch, ComBat and SVA we considered the following batch adjustment methods: mean-centering, standardization, ratio-A and ratio-G—as already mentioned in the section 'Background'.

Mean-centering (`meanc`)

From each measurement the mean of the values of the corresponding variable in the corresponding batch is substracted:

$$\widehat{x^*_{ijg}} = x_{ijg} - \widehat{\mu_{jg}}, \tag{18}$$

where $\widehat{\mu_{jg}} = (1/n_j)\sum_j x_{ijg}$.

Standardization (`stand`)

The values of each variable are centered and scaled per batch:

$$\widehat{x^*_{ijg}} = \frac{x_{ijg} - \widehat{\mu_{jg}}}{\sqrt{\widehat{\sigma^2_{jg}}}}, \tag{19}$$

where $\widehat{\mu_{jg}}$ as in (18) and $\widehat{\sigma^2_{jg}} = [1/(n_j - 1)]\sum_i (x_{ijg} - \widehat{\mu_{jg}})^2$.

Ratio-A (`ratioa`)

Each measurement is divided by the arithmetic mean of the values of the variable in the corresponding batch [15]:

$$\widehat{x^*_{ijg}} = \frac{x_{ijg}}{\widehat{\mu_{jg}}}, \tag{20}$$

where $\widehat{\mu_{jg}}$ is that same as in (18).

Ratio-G (`ratiog`)

Each measurement is divided by the geometric mean of the values of the variable in the corresponding batch [15]:

$$\widehat{x^*_{ijg}} = \frac{x_{ijg}}{\widehat{\mu_{g,geom}}}, \tag{21}$$

where $\widehat{\mu_{g,geom}} = \sqrt[n_j]{\prod_i^{n_j} x_{ijg}}$.

Addon adjustment of independent batches

As already described in the section 'Background', an important feature of batch effect adjustment methods is that they offer the possibility of making independent data more similar to already adjusted data of the same kind studying the same biological question of interest. This feature can be used for prediction purposes in particular. In the following we detail how batch effect adjustment is conceptionally performed for incorporating independent batches in general and for the particular methods considered in this paper.

General procedure

A batch effect adjustment method (implicitly or explicitly) assumes a specific model for the observed data. One part of parameters involved in this model is connected with the observed data within the batches x_{ijg} and another part with the unobserved batch effect free data x^*_{ijg}. While the values of the former kind of parameters in most cases depend on the individual batches, the latter kind are the same for all observations, i.e. these are batch-unspecific. When incorporating independent batches after having adjusted the training data, we are interested in transforming the data in the independent batches in such a way that its distribution becomes similar to that of the already adjusted training data without having to change the latter. This is achieved by performing the same kind of transformation on the independent batches with the peculiarity that for the involved batch-unspecific parameters the estimates obtained on the training data are used. We refer to these procedures as addon batch effect adjustment procedures.

Thus, for those batch effect adjustment methods, for which the corresponding adjustment does not involve estimated batch-unspecific parameters, the addon procedure is the same as the corresponding batch effect adjustment method. From the batch effect adjustment methods considered in this paper, this is the case for mean-centering, standardization, ratio-A and ratio-G. Here the batch effect adjustment is performed batch by batch. The adjustment according to ComBat, FAbatch and SVA, respectively, does by contrast involve estimated batch-unspecific parameters.

ComBat

For ComBat, Luo et al. [15] present the addon procedure for the situation of having only one batch in the training data, which is again considered in [20] under the designation M-ComBat. The adjustment with ComBat according to Eq. (4) involves estimates of the batch-unspecific parameters α_g, σ_g^2 and $\boldsymbol{\beta}_g$. The class values $\boldsymbol{a}_{ij}$ are unknown in the test data, wherefore, as already noted in the section 'ComBat `(comb)`', the term $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ is excluded from the adjustment. The addon procedure for ComBat now consists of performing the adjustment in Eq. (4), where $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ is excluded and α_g together with σ_g^2 have been estimated in the adjustment of the training data.

FAbatch

The adjustment with FAbatch involves estimates of the same batch-unspecific parameters as that with ComBat (according to Eq. (4)): α_g, σ_g^2 and $\boldsymbol{\beta}_g$. However, unlike in the adjustment with ComBat, in FAbatch the term $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ is considered additionally. This is achieved—roughly—by estimating $\mathbb{E}(\boldsymbol{a}_{ij}|x_{ij1},\ldots,x_{ijp})$ and $\boldsymbol{\beta}_g$ using L_2-penalized logistic regression. See again the section 'Estimation' for details. The addon procedure for FAbatch is straightforwardly derived from the general definition of addon procedures given above: the estimation scheme in the section 'Estimation' is performed with the peculiarity that for all occurring batch-unspecific parameters, the estimates obtained in the adjustment of the training data are used.

SVA

For SVA there exists a specific procedure denoted as "frozen SVA" [17], abbreviated as "fSVA," for preparing independent data for prediction as already mentioned in the section 'SVA `(sva)`'. More precisely, [17] describe two versions of fSVA, where one of these can be seen to be an addon procedure as defined above. The b_{gl}- and the $\boldsymbol{\beta}_g$-values are two of the batch-unspecific parameters involved in the SVA adjustment, see again the section 'SVA `(sva)`'. The $\boldsymbol{\beta}_g$-values are implicitly involved, namely when multiplying the variable values by the estimated probabilities that the corresponding variable is associated with unmeasured confounders, but not with the binary

variable representing the biological signal. In both frozen SVA algorithms, when adjusting for batch effects in new observations the estimates of the b_{gl}-values obtained on the training data are used. Also, for multiplying the variable values of a new observation by the estimated probabilities, both algorithms use the estimates obtained on the training data. The distinguishing feature between the two algorithms is the way estimates of the factors Z_{ijl} for new observations are obtained. In the section 'SVA `(sva)`' we stated that the factor estimates for SVA are obtained as the right singular vectors from a SVD.

In the first frozen SVA algorithm, denoted as "exact fSVA algorithm" in [17], the latent factor vector for a new observation is estimated in the following way: 1) Combine the training data with the values of the new observation and multiply by the probabilities estimated on the training data; 2) Re-perform the SVD on the combined data from 1) and use the right singular vector corresponding to the new observation as the estimate of its vector of latent factors. This algorithm is however not an addon procedure. In this algorithm, the estimate of the latent factor vector for the test observation originates from a different SVD than the estimated latent factors of the training observations. Therefore, this new estimated latent factor behaves—at least to some extent—differently than that of the training data. As a consequence, when adjusting the new observation a feature of addon procedures is not given: the same kind of transformation must be performed for independent batches. This problem can be assumed to have a lower impact for larger training datasets. Here the latent factor model estimated on the training data depends less on whether a single new observation is included into the SVD or not. A solution to the problem of differently behaving latent factor estimates in training and test data would be the following: for adjusting the training data use the estimates of the latent factors (and their loadings) obtained in the second SVD performed after including the test observation. This would, however, again not correspond to an addon procedure, because then the adjusted training data would be changed each time a new observation is included, which is not allowed as stated above.

The second frozen SVA algorithm, denoted as "fast fSVA algorithm" in [17] takes a different approach. Here, the SVD is not re-performed entirely on the combination of the training data and the new observation. Instead, one essentially performs a SVD for calculating the right singular vector corresponding to the new observation, in which the left singular vectors and singular values are fixed to the values of these parameters obtained in the SVA, which had been performed on the training data. Thus in this adjustment, it is taken into account that the left singular vectors and singular values are batch-unspecific parameters. The resulting estimated latent factor vector of the new observation behaves in the same way as that of the training data, because here it originates from the same SVD. This algorithm does

correspond to an addon procedure, because the same kind of transformation is performed for independent batches, i.e. observations in the SVA model, without the need to change the training data.

The fSVA algorithms appear intuitive at first sight. However, using the training data estimated factor loadings (and other informations in the case of the fast fSVA algorithm) requires that the same sources of heterogeneity are present in training and test data, which might not be true for a test data batch from a different source. Thus, frozen SVA is only fully applicable when training and test data are similar, as stated by Parker et al. [17]. Nevertheless in the section 'Application in cross-batch prediction' we apply it in cross-batch prediction to obtain indications on whether the prediction performance of classifiers might even deteriorate through the use of frozen SVA when training and test data are very different.

Above we have described addon procedures for the batch effect adjustment methods that are considered in this paper. However, using our general definition of addon procedures, such algorithms can readily be derived for other methods as well.

Comparison of FAbatch with existing methods

A comprehensive evaluation of the ability to adjust for batch effects of our method in comparison to its competitors was performed—using both simulated as well as real datasets. The simulation enables us to study the performance, subject to basic settings and to use a large number of datasets. Nevertheless simulated data can never capture all properties found in real datasets from the area of the application. Therefore, in addition, we studied 14 publicly available real datasets, each consisting of at least two batches.

The value of batch effect adjustment contains different aspects, which are connected with the adjusted data itself or with the results of certain analyses performed using the latter. Therefore, when comparing batch effect adjustment methods it is necessary to consider several criteria, where each is concerned with a certain aspect. We calculated seven different metrics measuring the performance of each batch effect adjustment method on each simulated and each real dataset.

In the following, we first outline the seven metrics considered in the comparison study described above. Subsequently, we introduce the simulation designs and give basic information on the real datasets. The results of these analyses are presented and interpreted in the section 'Ability to adjust for batch effects'.

Performance metrics

Here we detail the performance metrics used to assess batch effect adjustment. Several of them are, in their original form, restricted to the case of

only two batches. For datasets with more than two batches they are extended as follows: 1) Calculate the original metric for all possible pairs of batches; 2) Calculate the weighted average of the values in 1) with weights proportional to the sum of the sample sizes in the two respective batches.

Separation score `(sepscore)` We derived this metric from the mixture score presented in [11]. The latter was not applicable here, because it depends on the relative sizes of the two involved batches j and j^*. Roughly speaking the mixture score measures the degree of mixing between the observations belonging to the two batches after batch effect adjustment. The separation score by contrast measures the degree of separation between the two batches. At first for each observation in j, its k nearest neighbours are determined in both batches simultaneously with respect to the euclidean distance. Here, the proportion of those nearest neighbours belonging to batch j^* is calculated. Then the average—denoted as MS_j—is taken over the n_j proportions obtained in this way. This value is the mixture score as in [11]. To obtain a measure for the separation of the two batches the absolute difference between MS_j and its value expected in the absence of batch effects is taken: $|\,\mathrm{MS}_j - n_{j^*}/(n_j + n_{j^*} - 1)|$. The separation score is defined as the simple average of the latter quantity and the corresponding quantity when the roles of j and j^* are switched. The number k of nearest neighbours considered was set to 10. Smaller values of the separation score are better.

Average minimal distance to other batch `(avedist)` A very similar metric for two batches is the average minimal distance to the other batch after batch effect adjustment, see also [11]. For each observation in batch j the euclidean distance to the nearest observation in batch j^* is calculated. Consecutively the roles of j and j^* are switched and finally the average is computed over all $n_j + n_{j^*}$ minimal distances. To obtain a metric independent of the scale, we standardize the variables before the calculation to have zero mean and uniform variance. Here, smaller values are better.

Kullback-Leibler divergence between density of within and between batch pairwise distances `(klmetr)` This metric, used in [12] in a similar form is again based on the distances of the observations within and between batches. At first the distances between all pairs of observations in batch j—denoted as $\{dist_j\}$—and the distances between all such pairs in batch j^*—denoted as $\{dist_{j^*}\}$—are calculated. Then for each observation in j the distances to all observations in j^* are calculated, resulting in $n_j \times n_{j^*}$ distances denoted as $\{dist_{jj^*}\}$. Consecutively we estimate the Kullback-Leibler divergence between the densities of $\{dist_j\}$ and $\{dist_{jj^*}\}$ and that between the densities of $\{dist_{j^*}\}$ and $\{dist_{jj^*}\}$—using the k-nearest neigh-

bours based method by Boltz et al. [2] with $k = 5$. Finally, we take the weighted mean of the values of these two divergences with weights proportional to n_j and n_{j^*}. As in the case of `avedist` the variables are standardized before the calculation to make the metric independent of scale. Smaller values of this metric are better.

Skewness divergence score (`skewdiv`) This metric presented in [19] is concerned with the values of the skewness of the observation-wise empirical distributions of the data. Because batch effect adjustment should make the distribution of the data similar for all batches, these skewness values should not differ strongly across batches after a successful batch effect adjustment. The metric is obtained as follows for two batches j and j^* after batch effect adjustment: 1) for each observation calculate the difference between the mean and the median of the data as a measure for the skewness of the distribution of the variable values; 2) determine the area between the two batch-wise empirical cumulative density functions of the values out of 1). The value obtained in 2) can be regarded as a measure for the disparity of the batches with respect to the skewness of the observation-wise empirical distributions. Again, standardization is conducted before the calculation. Smaller values indicate a more successful batch effect adjustment with respect to the homogeneity of the skewness values.

Proportion of variation induced by class signal estimated by Principal Variance Components Analysis (`pvca`) Principal Variance Component Analysis [14] allows the estimation of the contributions of several sources of variability. Here, first principal component analysis is performed on the $n \times n$ covariance matrix between the observations. Then, using a random effects model, the principal components are regressed on arbitrary factors of variability, such as "batch" and "(phenotype) class". Ultimately, estimated proportions of variance induced by each factor, and that of the residual variance are obtained; for details see [14]. We included the factors "batch", "class" and the interaction of these two into the model and used the proportion of variance explained by "class" as a metric. Naturally, higher values of this metric indicate a better preservation or exposure, respectively, of the biological signal of interest.

Performance of differential expression analysis (`diffexpr`) This metric is similar to the idea presented in [11] which consists in comparing the list of genes deemed differentially expressed the strongest using a batch effect adjusted dataset to the corresponding list obtained using an independent dataset. Having no independent data available here we had to consider a slightly different approach: 1) For each batch j leave this batch out and perform batch effect adjustment using the rest of the dataset. Derive two

lists of the 5% of variables deemed differentially expressed the strongest (see next paragraph for details): one using the batch effect adjusted dataset—where batch j was left out—and one using the data from batch j. Calculate the number of variables appearing in both lists and divide this number by the common length of the lists. 2) Calculate a weighted average of the values obtained in 1) with weights proportional to the number of observations in the corresponding left-out batches. Note that in the case of the simulated datasets we would be able to estimate the true discovery rate instead of calculating the metric described above. However, for the sake of comparability, we applied the procedure described above for the simulated data as well.

Now we describe the procedure performed for estimating those 5% of variables which are most differentially expressed. Our original idea to use the p-values of simple two-sample t-tests between the two classes was soon discarded. The reason for this was that this procedure might have favoured batch effect adjustment methods that produce more normally distributed values of the variable. The p-values of classical non-parametric tests, such as the Mann-Whitney-Wilcoxon rank sum test would also not have been suitable here, because of the fact that here the p-values can only adopt a limited number of possible values. Therefore, it would have occurred in many cases that more than 5% of the variables adopt the smallest of possible p-values, making a selection of 5% of variables with the smallest p-values impossible. As a solution, for each variable we drew a randomized p-value out of the Whitney-Wilcoxon rank sum test, see [6] for details. These randomized p-values can adopt any possible value between zero and one and were consequently suitable for ordering the variables according to their degree of differential expression between the two classes. We ultimately considered those 5% variables that were associated with the smallest p-values. Higher values of this metric are better.

Mean Pearson's correlation of the variable values before and after batch effect adjustment `(corbeaf)` This metric suggested by Lazar et al. [11] is not a measure for the performance of batch effect adjustment. However, it may be used occasionally to decide between two methods performing similarly: in such cases the method that least affects the data—i.e. that with smaller `corbeaf`-values—could be preferred [11].

Simulation design

Three basic scenarios were considered: 1) "ComCor": Common correlation structure in all batches; 2) "BatchCor": Batch-specific correlation structures; 3) "BatchClassCor": Batch- and class-specific correlation structures. For each of these the correlations were induced in two ways (see below for details): 1) simulating from a latent factor model with normally distributed residuals; 2) drawing from multivariate normal distributions with specified

correlation matrices. The second scheme was considered to avoid artificially favouring FAbatch and SVA by restricting the simulation to factor-based data generation mechanisms. We simulated datasets consisting of four batches with 25 observations each. The number of variables was 1000. For each of the six (3×2) settings 500 datasets were simulated. The values of the parameters occurring in the simulation models were based on corresponding estimates obtained from two publicly available microarray datasets: a dataset also used in the real data study, denoted as `AutismTranscr` (Table 1) and a dataset studying colon cancer, denoted as `ColoncbTranscr`. The latter is downloadable from ArrayExpress [10], accession number: E-GEOD-44861.

All six settings can be expressed using the following most general model:

$$\begin{aligned} \boldsymbol{x}_{ij} &= \boldsymbol{\alpha} + \boldsymbol{a}_{ij}\boldsymbol{\beta} + \boldsymbol{\gamma}_j + \boldsymbol{\epsilon}^*_{ij}, \\ \boldsymbol{\epsilon}^*_{ij} &\sim MVN(\mathbf{0}, \boldsymbol{\Sigma}_{j,\boldsymbol{a}_{ij}}), \end{aligned} \tag{22}$$

with $\boldsymbol{x}_{ij} = (x_{ij1}, \ldots, x_{ijp})^T$, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_p)^T$, $\boldsymbol{a}_{ij} \in \{0, 1\}$, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$, $\boldsymbol{\gamma}_j = (\gamma_{j1}, \ldots, \gamma_{jp})^T$, $\boldsymbol{\epsilon}^*_{ij} = (\epsilon^*_{ij1}, \ldots, \epsilon^*_{ijp})^T$, $j \in \{1, \ldots, K\}$ and $p = 1000$.

The entries of $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}_j$ ($j \in \{1, \ldots, K\}$) were drawn from normal distributions with means and variances based on corresponding estimates obtained on `ColoncTranscr`. For details see the corresponding commented R code provided in Additional file 1. The vector of the class differences $\boldsymbol{\beta}$ contains 300 (30%) non-zero values. Half of these are negative and half positive. The values were drawn from gamma distributions, where the choice of parameters was again based on `ColoncTranscr`. Here, in the case of the negative entries of $\boldsymbol{\beta}$, the sign of the originally drawn values was changed.

The six settings differ with respect to the specification of $\boldsymbol{\Sigma}_{j,\boldsymbol{a}_{ij}}$. The differences are outlined in the following.

Design A: Simulating from latent factor model The residuals of the fixed part of the model $\boldsymbol{\epsilon}^*_{ij}$ were simulated in the following ways for the

corresponding scenarios:

1. ComCor:
$$\epsilon^*_{ijg} := \sum_{m=1}^{5} b_{0gm} Z_{ijm} + \delta_{jg}\epsilon_{ijg} \tag{23}$$

2. BatchCor:
$$\epsilon^*_{ijg} := \sum_{m=1}^{5} b_{0gm} Z_{ijm} + \sum_{m=1}^{5} b_{jgm} \overset{*}{Z}_{ijm} + \delta_{jg}\epsilon_{ijg} \tag{24}$$

3. BatchClassCor:
$$\epsilon^*_{ijg} := \sum_{m=1}^{5} b_{0gm} Z_{ijm} + \sum_{m=1}^{5} \widetilde{b}_{\boldsymbol{a}_{ij}gm} Z_{ijm} + \sum_{m=1}^{5} b_{jgm} \overset{*}{Z}_{ijm} + \delta_{jg}\epsilon_{ijg}, \tag{25}$$

where $\epsilon_{ijg} \overset{iid}{\sim} N(0, \sigma_g^2)$ and Z_{ijm}, $\overset{*}{Z}_{ijm} \overset{iid}{\sim} N(0,1)$. b_{0gm}, b_{jgm} and $\widetilde{b}_{\boldsymbol{a}_{ij}gm}$ are drawn from normal distributions and δ_{jg}^2 and σ_g^2 from inverse gamma distributions. The parameters of the latter distributions are again based on corresponding estimates obtained on `ColoncTranscr`.

In Eq. (23), (24) and (25) the factors $Z_{ij1}, \ldots, Z_{ij5}$ model the biological correlation between the variables. The factors $\overset{*}{Z}_{ij1}, \ldots, \overset{*}{Z}_{ij5}$ in (24) and (25) model distortions that affect the correlation in the batches. In setting "ComCor" all observations have the same correlation structure—independent of the batch. In setting "BatchCor" the correlation structure is different in each batch, due to the batch-specific loadings of the factors $\overset{*}{Z}_{ij1}, \ldots, \overset{*}{Z}_{ij5}$. In the third setting, "BatchClassCor", the correlations differ not only by batch but also according to which of the two classes the observations are in, i.e. we have batch- and class-specific correlations. In each setting the variances are different in the batches.

Design B: Drawing from multivariate distributions with specified correlation matrices All correlation matrices appearing in the three scenarios were estimated on real data. Here we first calculated the approximate positive definite correlation matrix using the R function `cor` and then applied the R function `nearPD` from the R package `Matrix` to the result to calculate the nearest positive definite correlation matrix. We used the 1000 genes from the `AutismTranscr` dataset, which showed themselves to be most related to the binary outcome according to variable-wise two-sample t-tests. Before estimating the correlation matrices, the data was further centered by

class in each batch to adjust for excess correlations due to class differences. The variances are the same in all three scenarios. They were set to be equal to those in scenario ComCor of Design A, i.e. $\sum_{m=1}^{5} b_{0gm}^2 + \delta_{jg}^2 \sigma_g^2$.

The correlation matrices were obtained as follows for the three scenarios:

1. ComCor: A single correlation matrix was used for all batches here. It was estimated from the data of a single batch.

2. BatchCor: A separate correlation matrix was used for each batch here, each estimated from the data of a batch in `AutismTranscr`.

3. BatchClassCor: A separate correlation matrix was used for each combination of batch and class here, where each was estimated on a corresponding batch-class-combination in `AutismTranscr`.

After obtaining the correlation matrices, the corresponding covariance matrices were calculated. The latter was done by multiplying each entry in the correlation matrices with the respective pair of standard deviations.

Datasets

We used 14 high-dimensional datasets with a binary target variable and at least two batches. They were downloaded from the ArrayExpress database (`www.ebi.ac.uk/arrayexpress`) [10] or the NCBI GEO database (`www.ncbi.nlm.nih.gov/geo`)
[1]. Table 1 gives an overview on the datasets.

For details the reader may look up the accession numbers online and consult the corresponding R scripts written for preparation of the datasets, which are available in Additional file 1. Here we also provide all R code necessary to reproduce our analyses.

Results

Ability to adjust for batch effects

Supplementary Figures 1 to 7 (Additional file 2) show the values of the individual metrics obtained on the simulated data and Figure 2 shows the corresponding results obtained on the 14 real datasets. Supplementary Tables 1 to 7 (Additional file 2) for the simulated and Tables 2 and 3 for the real data, respectively show the means of the metric values separated by method (and simulation scenario) together with the mean ranks of the methods with respect to the individual metrics. In most cases, we observe that the simulation results differ only slightly between the settings with respect to the ranking of the methods by their performance. Therefore, we will only occasionally differentiate between the scenarios in the interpretations. Similarly,

Label	Num. of observ.	Num. of batches	Num. of variables	Prop. with y = 2	Data type	Source (Acc.num.)
ColonGastricEsophagealcSNPArray	93	3	50000	0.54	comparative genomic hybridization	ArrayExpr.: E-GEOD-36458
AgeDichotomTranscr	243	15	27568	0.49	DNA methylation profiling	ArrayExpr.: E-GEOD-36194
EthnicityMethyl	133	3	50000	0.45	DNA methylation profiling	ArrayExpr.: E-GEOD-39672
BipolardisorderMethyl	94	2	27537	0.50	DNA methylation profiling	ArrayExpr.: E-GEOD-38873
PostpartumDepressionMethyl	50	5	50000	0.46	DNA methylation profiling	ArrayExpr.: E-GEOD-44132
AutismTranscr	439	5	24526	0.53	transcription profiling	ArrayExpr.: E-GEOD-37772
BreastcTranscr	410	23	20180	0.50	transcription profiling	ArrayExpr.: E-GEOD-44281
BreastCancerConcatenation	168	5	22277	0.65	transcription profiling	ArrayExpr.: E-GEOD-27562, E-GEOD-21422, E-GEOD-22544, E-GEOD-20266, E-TABM-276
IUGRTranscr	67	2	48701	0.40	transcription profiling	ArrayExpr.: E-GEOD-35574
IBSTranscr	63	6	54671	0.70	transcription profiling	ArrayExpr.: E-GEOD-36701
SarcoidosisTranscr	58	3	54675	0.66	transcription profiling	NCBI GEO: GSE19314
pSSTranscr	49	3	54675	0.63	transcription profiling	ArrayExpr.: E-GEOD-40611
AlcoholismTranscr	39	2	28869	0.51	transcription profiling	ArrayExpr.: E-GEOD-44456
WestNileVirusTranscr	39	2	47323	0.46	transcription profiling	ArrayExpr.: E-GEOD-43190

Table 1 Overview of datasets used in empirical studies. The following information is given: number of observations, number of batches, number of variables, proportion of observations with disease, biomolecular data type, accession number

simulations and real data analyses often yield similar results. Differences will be discussed whenever relevant.

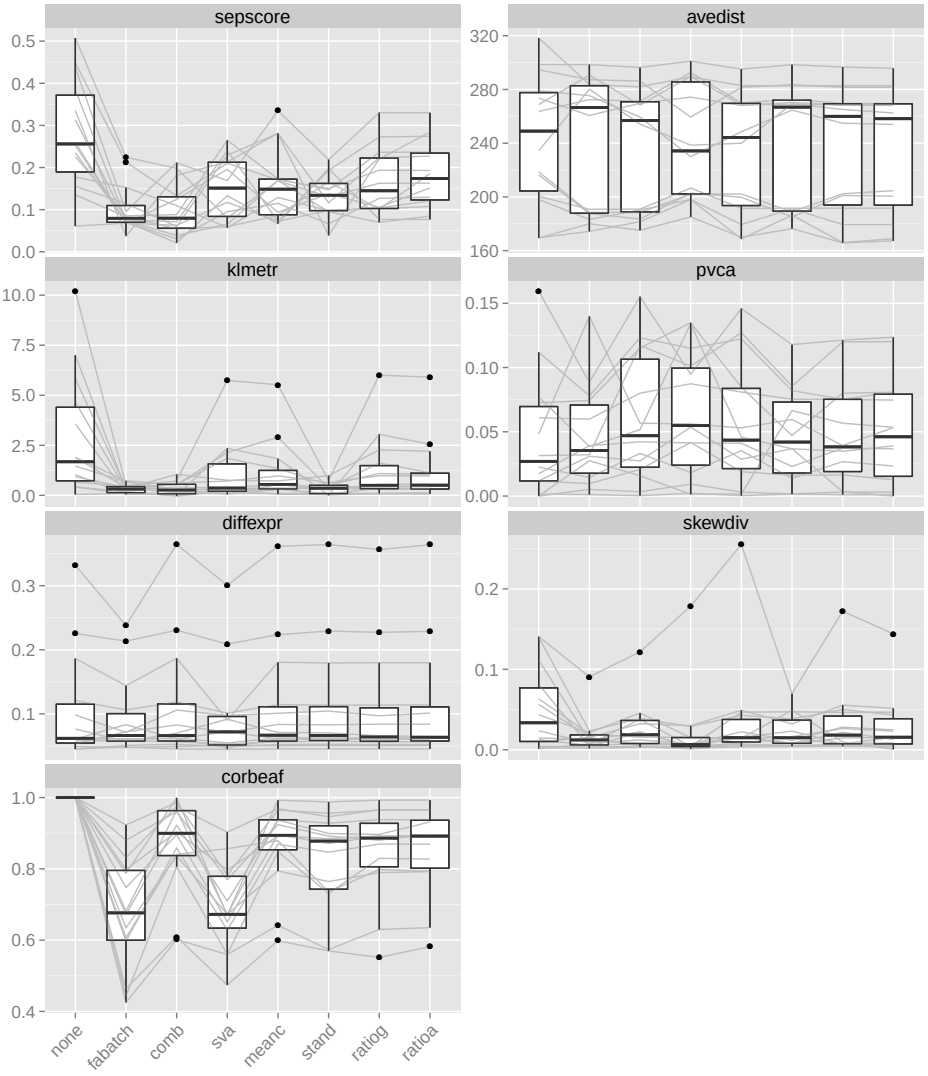

Figure 2 Metric values in real datasets. Boxplots of values for all 14 datasets separated into method for the following metrics: `sepscore`, `avedist`, `klmetr`, `pvca`, `diffexpr`, `skewdiv` and `corbeaf`. The grey lines connect values corresponding to the same datasets.

According to the values of the separation score (Supplementary Figure 1 and Figure 2, Supplementary Table 1 and Table 2) ComBat, FAbatch and standardization seem to lead to the best mixing of the observations across the batches. For the real datasets, however, standardization was only slightly better on average than other methods.

sepscore

mean values	comb 0.09895	**fabatch** **0.10227**	stand 0.13238	sva 0.15217	meanc 0.15807	ratiog 0.16618	ratioa 0.18314	none 0.2806
mean ranks	comb 2.28571	**fabatch** **3.35714**	stand 3.71429	sva 4.42857	meanc 4.64286	ratiog 4.78571	ratioa 5.92857	none 6.85714

avedist

mean values	meanc 233.32619	ratiog 235.27321	ratioa 235.39525	comb 235.52757	stand 237.86855	**fabatch** **239.53197**	sva 240.55365	none 243.10948
mean ranks	meanc 3.07143	comb 3.57143	ratiog 3.57143	ratioa 3.57143	**fabatch** **5.14286**	stand 5.21429	none 5.78571	sva 6.07143

klmetr

mean values	**fabatch** **0.32312**	comb 0.33748	stand 0.35524	sva 1.08835	meanc 1.13029	ratioa 1.15025	ratiog 1.23577	none 2.85956
mean ranks	comb 2.85714	**fabatch** **3**	stand 3.14286	sva 4.57143	meanc 4.71429	ratioa 4.92857	ratiog 5.42857	none 7.35714

pvca

mean values	sva 0.06364	comb 0.06015	meanc 0.05636	ratioa 0.0502	ratiog 0.04933	stand 0.04741	**fabatch** **0.04569**	none 0.04477
mean ranks	sva 2.92857	comb 3.14286	meanc 3.57143	ratioa 5	stand 5.07143	ratiog 5.21429	**fabatch** **5.35714**	none 5.71429

Table 2 Means of the metric values and of their ranks among the different methods over the 14 studied datasets separated into method for the following metrics: `sepscore`, `avedist`, `klmetr` and `pvca`. In each row the results are listed in descending order according to mean performance in terms of the original values and their ranks, respectively.

diffexpr

mean values	comb 0.11044	stand 0.10958	ratioa 0.10891	meanc 0.1088	ratiog 0.10796	none 0.10526	sva 0.09517	**fabatch** **0.09364**
mean ranks	comb 3.28571	stand 3.57143	ratioa 3.78571	meanc 4.14286	none 4.5	ratiog 4.64286	**fabatch** **5.85714**	sva 6.21429

skewdiv

mean values	**fabatch** **0.01724**	sva 0.02206	stand 0.02377	comb 0.02688	ratioa 0.02875	ratiog 0.03257	meanc 0.03671	none 0.05041
mean ranks	sva 2.21429	**fabatch** **2.92857**	comb 4.28571	stand 4.78571	meanc 5.07143	ratioa 5.42857	ratiog 5.5	none 5.78571

corbeaf

mean values	none 1	comb 0.86857	meanc 0.86742	ratioa 0.85516	ratiog 0.84931	stand 0.82754	sva 0.69313	**fabatch** **0.67795**
mean ranks	none 1	comb 2.92857	meanc 2.92857	ratiog 4.21429	ratioa 4.35714	stand 5.85714	sva 7.14286	**fabatch** **7.57143**

Table 3 Means of the metric values and of their ranks among the different methods over the 14 studied datasets separated into method for the following metrics: `diffexpr`, `skewdiv` and `corbeaf`. In each row the results are listed in descending order according to mean performance in terms of the original values and their ranks, respectively.

The results with respect to `avedist` are less clear. The simulation with factors (Design A) suggests that FAbatch and SVA are associated with greater minimal distances to neighboring batches, compared to the other methods. However, we do not clearly observe this for Design B other than for the setting with common correlations. The real data results also suggest no clear ordering between the methods with respect to this metric; see in particular the means over the datasets in Table 2. The values of this metric were not appreciably improved by batch effect adjustment in general on the real datasets.

The values of `klmetric`, which is conceptionally very similar to the separation score, allows a very similar conclusion as the latter metric (Supplementary Figure 3 and Figure 2, Supplementary Table 3 and Table 2): ComBat, FAbatch and standardization performed best here. While this conclusion could be obtained on both simulated and real data, other results differed between the different simulation scenarios and the real data analyses: SVA performed considerably worse here for Design A than B and mean-centering performed better on the simulated data in general.

The estimates of the proportions of the variation explained by the class signals obtained via Principal Variance Components Analysis (`pvca`) are depicted in the Supplementary Figure 4 and Figure 2 and summarized in the Supplementary Table 4 and Table 2. SVA appears to be associated with the highest proportion of variation induced by the class signal. However, the comparison to the other methods is not fair here: SVA makes use of the target variable and is therefore associated with an artificially increased class signal. See the section 'Application in cross-batch prediction' for details on this mechanism related to overoptimism. FAbatch performed well only on the simulated data here, but not on the real datasets, where it had the lowest mean value with the exception of no batch effect adjustment. Figure 2 reveals that those three datasets for which `pvca` was considerably smaller after batch effect adjustment by FAbatch were, at the same time, the three datasets with the highest `pvca`-values before batch effect adjustment. Datasets with high `pvca`-values are datasets where the biological signal is relatively strong in comparison to the batch effects. Our results suggest that for such datasets, batch effect adjustment with FAbatch might be counterproductive. The distinguishing feature of FAbatch in comparison to a mere location-scale adjustment as performed by ComBat is that it aims at additionally adjusting for batch effects not explainable by location and scale shifts. While FAbatch aims at protecting the biological signal in the factor estimation, it cannot be protected entirely here due to the uncertainty in the estimation of the class probabilities. When reducing the total heterogeneity by FAbatch in cases of weak batch effects, the merit of removing heterogeneity due to batch effects becomes smaller in comparison to the harm that affects the signal. ComBat performed better than other methods here on the real data (with the exception of SVA as mentioned before).

For the performance metric related to differential expression analysis `diffexpr` (Supplementary Figure 5 and Figure 2, Supplementary Table 5 and Table 3) the results for FAbatch and SVA are quite different between simulated and real data. In the simulation, the two methods performed best compared to the others (with the exception of FAbatch for Design B with common correlation). However, for the real data they performed worst—even worse than no batch effect adjustment in the mean. For FAbatch we examined those datasets which yielded substantially worse `diffexpr`-values after batch effect adjustment than before. As can already be seen from Figure 2, two of these datasets have high `diffexpr`-values on the data before batch effect adjustment. This implies that for these datasets the biological signal is well preserved in the batches—in other words they seem to be less affected by batch effects. A possible reason why FAbatch performs worse for mild batch effects has already been outlined above. The other datasets connected with worse `diffexpr`-values than "no batch effect adjustment" in the case of FAbatch were those datasets for which some "outlying" batches were very different from the others—according to the PCA plots given in Supplementary Figure 8 (Additional file 2). We conjecture that, in this case, pooling the data of the outlying batch(es) with the other batches and estimating the L_2-penalized logistic regression model can result in a predictor with bad performance. The combined data might be too heterogeneous for the L_2-penalized logistic regression model, which assumes that all observations follow the same distribution. If the predictions of the class probabilities by the L_2-penalized logistic regression rule are bad, the biological signal is less protected in the latent factor estimation. Therefore, the removal of the estimated latent factor influences will affect the biological signal more. There were no noteworthy differences between the other methods with respect to `diffexpr`. For the real datasets there were also no improvements over no batch effect adjustment. This indicates that differential expression analysis might not benefit from batch effect adjustment in general.

For the skewness divergence score (Supplementary Figure 6 and Figure 2, Supplementary Table 6 and Table 3) no clear ranking between the methods is seen in the case of the simulated data. However, for the real datasets, SVA and FAbatch clearly outperform the other methods with respect to this metric.

Finally, both in the simulated and real data, FAbatch and SVA have considerably lower `corbeaf`-values (Supplementary Figure 7 and Figure 2, Supplementary Table 7 and Table 3), which is not very surprising considering their high complexity.

Application in cross-batch prediction

In this illustrative analysis we apply all batch effect adjustment methods considered above together with the corresponding addon procedures described

in the section 'Addon adjustment of independent batches' in a cross-batch prediction example using a real dataset. For an extensive real data study see [15], who use several datasets to compare all of the methods considered here, except for frozen SVA ("fSVA") and FAbatch, with respect to their performance in cross-batch prediction.

We use the dataset `IUGRTranscr` in this analysis. The reasons for choosing this dataset were that it features a relatively strong class signal and is at the same time strongly affected by batch effects—judging from the principal component analysis plot in Supplementary Figure 8. This dataset contains miRNA-measurements obtained from 67 human placentas using the Illumina Human-6 v2 Expression BeadChip. Of these 67 samples, 27 were obtained from placentas of embryos suffering from intrauterine growth restriction (IUGR), the remaining 40 samples originate from placentas of healthy embryos. The dataset consists of two batches of sizes 20 and 47, where in the first batch 9 (45%) and in the second batch 18 ($\approx 38\%$) samples originate from IUGR embryos.

As classification algorithm for the dependent variable "IUGR (yes vs. no)" Linear Discriminant Analysis (LDA) using Partial Least Squares (PLS) components as covariates [3] was chosen, where the number of components used was tuned on the grid $1, 2, \ldots, 10$ employing 3-fold CV.

Just as Luo et al. [15] in their extensive real data study, we use Matthews Correlation Coefficient (MCC) as performance metric. This measure has the advantage over the more commonly considered misclassification error rate, that it is independent of the class frequencies in the test data. It takes values in $[-1, 1]$, where a MCC-value of 1 would indicate a perfect prediction, a MCC-value of 0 would correspond to a completely random prediction and a MCC-value of -1 to a total disagreement between prediction and reality.

Figure 3 depicts the MCC-values resulting when applying the different batch effect adjustment methods in predicting from one batch to the other and than switching the training and test set roles between the two batches. When training on the first batch only ComBat, mean-centering and FAbatch lead to a higher MCC-value in comparison to no batch effect adjustment. The two fSVA algorithms and standardization lead to a very strong deterioration of the prediction performance, where the fast fSVA algorithm was slightly better than the exact fSVA algorithm. When training on the second batch, the prediction performance without batch effect adjustment corresponded to random guessing as indicated by the MCC-value of zero here. Except for standardization and the exact fSVA algorithm, all methods lead to a more or less strong improvement of prediction performance here. The ranking between the methods is almost entirely the same compared to that when training on the first batch. Note again that this analysis is only illustrative. Nevertheless, the severity of the deterioration of the prediction performance when applying fSVA (and standardization) in the case of training on the first batch was striking. Considering that fSVA did not perform

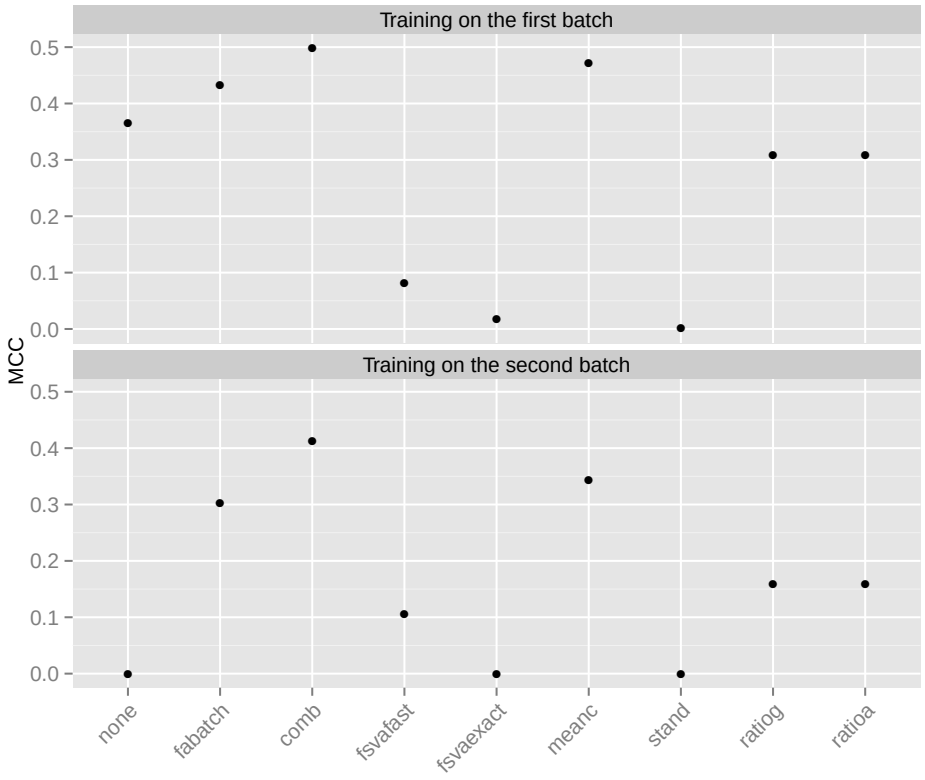

Figure 3 Cross-batch prediction—MCC-values. MCC-values out of using the individual batch effect adjustment methods in cross-batch prediction when training on the first and second batch. `fsvafast` and `fsvaexact` denote the fast and the exact fSVA algorithm, respectively.

that poorly when training on the second batch, a possible explanation for this strong deterioration might be the following: The first batch is much smaller than the second batch. In the section 'FAbatch `(fabatch)`' we detailed why using the actual values of the target variable in protecting the biological signal during the latent factor estimation of FAbatch would lead to an artificially increased class signal. SVA does use the values of the target variable and indeed suffers from the problem of an artificially increased class signal. The severity of this is more pronounced for smaller datasets. In the following, we will outline the reason why SVA suffers from this problem. A crucial problem with the weighting of the variable values by the estimated probabilities that the corresponding variable is associated with unmeasured confounders but not with the target variable is the following: these estimated probabilities depend on the values of the target variable, in particular for smaller datasets. Due to the variability in the data, for some variables the measurements are, by chance, separated overly strong between the two classes. Such variables, for which the observed separation between the classes is larger than the actual—biologically motivated—separation, are connected with smaller estimated weights. This means that such variables are affected less strongly by the removal of the estimated latent factor influences compared to variables which are not connected with such a randomly increased separation. As a result, after applying SVA the classes are separated to a stronger degree than they would be if biological differences between the classes were the only source of separation—as is required in a meaningful analysis.

The observed deterioration of the MCC-values by performing frozen SVA when training on the smaller batch may, admittedly, also be due to random error. In order to investigate whether the effects originating from the mechanism of artificially increasing the discriminative power of datasets by performing SVA are strong enough to have actual implications in data analysis, we performed a small simulation study. We generated datasets with 40 observations, 1000 variables, two equally sized batches, standard normally distributed variable values and a binary target variable with equal class probabilities. Note that there is no class signal in this data. Then using 5-fold cross-validation repeated two times we estimated the misclassification error rate of PLS followed by LDA for this data. Consecutively, we applied SVA to this data and again estimated the misclassification error rate of PLS followed by LDA using the same procedure. We repeated this procedure for the number of factors to estimate set to 1, 2 and 3, respectively. In each case we simulated 50 datasets. The mean of the misclassification error rates was 0.504 for the raw datasets and 0.431, 0.356 and 0.306 after applying SVA with 1, 2 and 3 factors. These results confirm that the artificial increase of the class signal by performing SVA can be strong enough to have implications in data analysis. Moreover, the problem seems to be more severe for a higher number of factors estimated. We did the same analysis with FAbatch,

again using 1, 2 and 3 factors, where we obtained the misclassification error rates 0.505, 0.521 and 0.509, respectively, suggesting that FAbatch does not suffer from this problem in the investigated context.

Discussion

In this paper, with FAbatch, we introduced a very general batch effect adjustment method for situations in which the batch membership is known. It accounts for two kinds of batch effects simultaneously: 1) coarse, easily observable batch effects expressed as location and scale shifts of the variable values across the different batches; 2) more complicated batch effects, modelled by latent factor influences, which affect the correlations between the variables in the batches. The model behind FAbatch is an extension of the model underlying ComBat, where the latter is designed to address the first kind of the batch effects described above. FAbatch uses latent factors to model batch effects in the spirit of SVA. In contrast to SVA, however, FAbatch assumes that the batch membership of the observations is known and that the latent factor models are batch-specific, i.e. that in each batch different sources of heterogeneity may operate. We saw in the section 'SVA `(sva)`' that in the SVA model it is implicitly assumed that the distribution of the vector of latent factors may be different for each observation. This is a very general assumption. However, it is unclear how well SVA can deal with specific datasets originating from such a general model, because the link between the singular value decomposition used in the estimation and this model is not evident. Our algorithm by contrast was explicitly motivated by its underlying model, which is quite general and reasonable. In cases in which the data in question is approximately uniform with this model, FAbatch should perform reasonably well. A further peculiarity of FAbatch is that—in contrast to SVA—it avoids generating a severe artificial increase in the biological signals of interest. This is achieved by not using the true classes of the observations when protecting the biological signal of interest in the factor estimation. In the form presented here, FAbatch is only applicable in the presence of a binary target variable. However, it can also be extended to other types of target variables. For example, when having a metric target variable one could use ridge regression instead of L_2-penalized logistic regression when protecting the biological signal of interest in the factor estimation.

In an illustrative analysis we applied the batch effect adjustment methods studied in the main analyses in the important case of cross-batch prediction. FAbatch—other than fSVA—performed reasonably well in this example. Moreover, by a small simulation study we obtained evidence that the artificial increase of the measured biological signal of interest faced when performing SVA can have noticeable negative effects in applications. In

FAbatch, this artificial increase is prevented by employing the following idea: for each observation the parameters involved in the transformations performed for protecting the biological signal are estimated using training data, which does not contain the respective observation to be transformed. This idea may also be applied in the protection of the biological signal of SVA, i.e. when multiplying the variable values by the estimated probabilities that the corresponding variables are associated with unmeasured confounders, but not with the binary variable representing the biological signal. More precisely these probabilities could be estimated in a cross-validation procedure—taking up again the idea already used in FAbatch.

All batch effect adjustment methods considered in this paper, together with the corresponding addon procedures and all metrics used in the comparisons of the methods, were implemented/adopted into the new R package `bapred` available online from CRAN [7].

Conclusions

FAbatch leads to a good mixing of the observations across the batches in comparison to other methods, which is reassuring given the diversity of batch effect structures in real datasets. In the case of very weak batch effects and in the case of strongly outlying batches, the observed biological signal may be slightly altered by FAbatch. In our extensive comparison study of existing and new batch effect correction methods, we found that no method was best with respect to all metrics. It is thus difficult to formulate general recommendations: the choice of the method may primarily depend on the goal of the researcher as reflected by the choice of the metric. Performing no batch effect correction at all is in any case not recommended.

Abbreviations `avedist`: average minimal distance to other batch; BatchClassCor: batch- and class-specific correlation structures; BatchCor: batch-specific correlation structures; ComCor: common correlation structure in all batches; `corbeaf`: mean Pearson's correlation of the variable values before and after batch effect adjustment; `diffexpr`: performance of differential expression analysis; fSVA: frozen SVA; `fsvaexact`: exact fSVA algorithm; `fsvafast`: fast fSVA algorithm; `klmetr`: Kullback-Leibler divergence between density of within and between batch pairwise distances; `pvca`: proportion of variation induced by class signal estimated by Principal Variance Components Analysis; `sepscore`: separation score; `skewdiv`: skewness divergence score; SVD: singular value decomposition.

Conflict of Interest The authors declare that they have no conflict of interest.

Acknowledgements We thank Sarah Tegenfeldt for making valuable language corrections. This work was supported by the German Science Foundation (DFG-Einzelförderung BO3139/3-1 to Anne-Laure Boulesteix).

References

[1] T Barrett, S E Wilhite, P Ledoux, C Evangelista, I F Kim, M Tomashevsky, K A Marshall, K H Phillippy, P M Sherman, M Holko, A Yefanov, H Lee, N Zhang, C L Robertson, N Serova, S Davis, and A Soboleva. Ncbi geo: archive for functional genomics data sets–update. *Nucleic Acids Res.*, 41:D991–D995, 2013.

[2] S Boltz, E Debreuve, and M Barlaud. High-dimensional statistical measure for region-of-interest tracking. *Transactions in Image Processing*, 18(6):1266–1283, 2009.

[3] A-L Boulesteix. PLS dimension reduction for classification with microarray data. *Stat. Appl. Genet. Mol. Biol.*, 3(1):33, 2004.

[4] C Chen, K Grennan, J Badner, D Zhang, E Gershon, L Jin, and C Liu. Removing batch effects in analysis of expression microarray data: An evaluation of six batch adjustment methods. *PLoS ONE*, 6(2):e17238, 2011.

[5] C Friguet, M Kloareg, and D Causeur. A factor model approach to multiple testing under dependence. *J. Am. Stat. Assoc.*, 104(488):1406–1415, 2009.

[6] C J Geyer and G D Meeden. Fuzzy and randomized confidence intervals and p-values (with discussion). *Stat. Sci.*, 20(4):358–387, 2005.

[7] R Hornung and D Causeur. *bapred: Batch Effect Removal (in Phenotype Prediction using Gene Data)*, 2015. R package version 0.1.

[8] C-W Hsu, C-C Chang, and C-J Lin. A practical guide to support vector classification. Technical report, National Taiwan University, 2010.

[9] W E Johnson, A Rabinovic, and C Li. Adjusting batch effects in microarray expression data using empirical bayes methods. *Biostatistics*, 8:118–127, 2007.

[10] N Kolesnikov, E Hastings, M Keays, O Melnichuk, Y A Tang, E Williams, M Dylag, N Kurbatova, M Brandizi, T Burdett, K Megy, E Pilicheva, G Rustici, A Tikhonov, H Parkinson, R Petryszak, U Sarkans, and A Brazma. ArrayExpress update – simplifying data submissions. *Nucleic Acids Res.*, 2015.

[11] C Lazar, S Meganck, J Taminau, D Steenhoff, A Coletta, C Molter, D Y Weiss-Solís, R Duque, H Bersini, and A Nowé. Batch effect removal methods for microarray gene expression data integration: a survey. *Briefings in Bioinformatics*, 14(4):469–490, 2012.

[12] J A Lee, K K Dobbin, and J Ahn. Covariance adjustment for batch effect in gene expression data. *Stat. Med.*, 33:2681–2695, 2014.

[13] J T Leek and J D Storey. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genetics*, 3:1724–1735, 2007.

[14] J Li, P Bushel, Chu T-M, and R D Wolfinger. Principal variance components analysis: Estimating batch effects in microarray gene expression data. In A Scherer, editor, *Batch Effects and Noise in Microarray Experiments: Sources and Solutions*, pages 141–154. John Wiley & Sons, Chichester, UK, 2009.

[15] J Luo, M Schumacher, A Scherer, D Sanoudou, D Megherbi, T Davison, T Shi, W Tong, L Shi, H Hong, C Zhao, F Elloumi, W Shi, R Thomas, S Lin, G Tillinghast, G Liu, Y Zhou, D Herman, Y Li, Y Deng, H Fang, P Bushel, M Woods, and J Zhang. A comparison of batch effect removal methods for enhancement of prediction performance using maqc-ii microarray gene expression data. *The Pharmacogenomics Journal*, 10:278–291, 2010.

[16] J N S Matthews. *Introduction to Randomized Controlled Clinical Trials*. Chapman & Hall, London, UK, 2006.

[17] H S Parker, H C Bravo, and J T Leek. Removing batch effects for prediction problems with frozen surrogate variable analysis. *PeerJ*, 2:e561, 2014.

[18] D B Rubin and D T Thayer. EM algorithms for ML factor analysis. *Psychometrika*, 47(1):69–76, 1982.

[19] A A Shabalin, H Tjelmeland, C Fan, C M Perou, and A B Nobel. Merging two gene-expression studies via cross-platform normalization. *Bioinformatics*, 24(9):1154–1160, 2008.

[20] C K Stein, P Qu, J Epstein, A Buros, A Rosenthal, J Crowley, G Morgan, and B Barlogie. Removing batch effects from purified plasma cell gene expression microarrays with modified combat. *BMC Bioinformatics*, 16:63, 2015.