Removing batch effects for prediction problems with frozen surrogate variable analysis

Hilary S. Parker[1], Héctor Corrada Bravo[2] and Jeffrey T. Leek*[1]

[1]Department of Biostatistics, Johns Hopkins Bloomberg School of Public Health, Baltimore, MD, 21205
[2]Center for Bioinformatics and Computational Biology, Department of Computer Science, University of Maryland, College Park, Maryland, 20742

June 20, 2014

Abstract

Batch effects are responsible for the failure of promising genomic prognostic signatures, major ambiguities in published genomic results, and retractions of widely-publicized findings. Batch effect corrections have been developed to remove these artifacts, but they are designed to be used in population studies. But genomic technologies are beginning to be used in clinical applications where samples are analyzed one at a time for diagnostic, prognostic, and predictive applications. There are currently no batch correction methods that have been developed specifically for prediction. In this paper, we propose an new method called frozen surrogate variable analysis (fSVA) that borrows strength from a training set for individual sample batch correction. We show that fSVA improves prediction accuracy in simulations and in public genomic studies. fSVA is available as part of the `sva` *Bioconductor* package. genomics; prediction; batch effects; personalized medicine; surrogate variable analysis

*jleek@jhsph.edu

1 Introduction

Genomic technologies were originally developed and applied for basic science research and hypothesis generation (Eisen *et al.*, 1998). As these technologies mature, they are increasingly being used as clinical tools for diagnosis or prognosis (Chan and Ginsburg, 2011). The high-dimensional measurements made by microarrays can be used to classify patients into predictive, prognostic, or diagnostic groups. Despite the incredible clinical promise of these technologies there have only been a few signatures that have successfully been translated into the clinic.

One of the reasons for the relatively low rate of success is the impact of unmeasured technological or biological confounders. These artifacts are collectively referred to as "batch effects" because the processing date, or batch, is the most commonly measured surrogate for these unmeasured variables in genomic studies (Scharpf *et al.*, 2011; Johnson *et al.*, 2007; Walker *et al.*, 2008). The umbrella term batch effects also refers to any unmeasured variables that can vary from experiment to experiment, ranging from the technician who performs the experiment to the temperature and ozone levels that day (Lander, 1999; Fare *et al.*, 2003).

Batch effects are responsible for the failure of promising genomic prognostic signatures (Baggerly *et al.*, 2004, 2005), major ambiguities in published genomic results (Spielman *et al.*, 2007; Akey *et al.*, 2007), and retractions of widely-publicized findings (Sebastiani *et al.*, 2010; Lambert and Black, 2012). In many experiments, the signal from these unmeasured confounders is larger than the bi-

ological signal of interest (Leek *et al.*, 2010). But the impact of batch effects on prediction problems has only recently been demonstrated (Parker and Leek, 2012; Luo *et al.*, 2010). Batch effects were also recognized as a significant hurdle in the development of personalized genomic biomarkers in the Institute of Medicine's report on clinical genomics (Micheel *et al.*, 2012).

While a number of methods have been developed for removing batch effects in population-based genomic studies (Johnson *et al.*, 2007; Gagnon-Bartsch and Speed, 2012; Leek and Storey, 2007; Leek *et al.*, 2010; Walker *et al.*, 2008), there is currently no method for removing batch effects for prediction problems. There are two key differences between population level corrections and corrections designed for prediction problems. First, population level corrections assume that the biological groups of interest are known in advance. In prediction problems, the goal is to predict the biological group. Second, in prediction problems, new samples are observed one at a time, so the surrogate batch variable will have a unique and unknown value for each sample.

Here we propose frozen Surrogate Variable Analysis (fSVA) as a method for batch correction in prediction problems. fSVA borrows strength from a reference database to address the challenges unique to batch correction for prediction. The fSVA approach has two main components. First, surrogate variable analysis (SVA) is used to correct for batch effects in the training database. Any standard classification algorithm can then be applied to build a classifier based on this clean training data set. Second, probability weights and coefficients estimated on the training database are used to remove batch effects in new samples. The classifier

trained on the clean database can then be applied to these cleaned samples for prediction.

We show with simulated data that the fSVA approach leads to substantial improvement in predictive accuracy when unmeasured variables are correlated with biological outcomes. We also apply fSVA to multiple publicly available microarray data sets and show improvements in prediction accuracy after correcting for batch. The methods developed in this paper have been implemented in the freely available `sva` Bioconductor package.

2 Frozen Surrogate Variable Analysis methodology

2.1 Removing batch effects from the training set

The first step in batch correction for prediction problems is to remove batch effects from the training set. In the training set, the biological groups are known. This setting is similar to the population genomics setting and we can use a model for gene expression data originally developed for population correction of unmeasured confounders. If there are m measured features and n samples in the training set, we let $X_{m\times n}$ be the matrix of feature data, where x_{ij} is the value of feature i for sample j. For convenience, we will refer to X as the expression matrix for the remainder of the paper. However, our methods can be generally applied to any set of features, including measures of protein abundance, gene expression, or DNA methylation.

We propose a linear model for the relationship between the expression levels

and the outcome of interest y_j: $x_{ij} = b_0 + \sum_{k=1}^{p_1} s_k(y_j) + e_{ij}$. The $s_k(\cdot)$ are a set of basis functions parameterizing the relationship between expression and outcome. If the prediction problem is two-class, then $p_1 = 1$ and $s_1(y_j) = 1(y_j = 1)$ is an indicator function that sample j belongs to class one. In a multi-class prediction problem $k > 1$ and the $s_k(\cdot)$ may represent a factor model for each class. In matrix form this model can be written (Leek and Storey, 2007, 2008).

$$X = BS + E \tag{1}$$

where $S_{p_1 \times n}$ is a model matrix of p_1 biological variables of interest for the n samples, $B_{m \times p_1}$ are the coefficients for these variables, and $E_{m \times n}$ is the matrix of errors.

In genomic studies, the error term E is not independent across samples (Johnson *et al.*, 2007; Leek and Storey, 2007, 2008; Walker *et al.*, 2008; Friguet *et al.*, 2009; Leek *et al.*, 2010; Gagnon-Bartsch and Speed, 2012). That is, there is still correlation between rows of E after accounting for the model S. The correlation is due to unmeasured and unwanted factors such as batch. We can modify model 1 to account for these measured biological factors and unmeasured biological and non-biological factors:

$$X = BS + \Gamma G + U \tag{2}$$

where $G_{p_2 \times n}$ is a $p_2 \times n$ random matrix, called a dependence kernel (Leek and Storey, 2008) that parameterizes the effect of unmeasured confounders, $\Gamma_{m \times p_2}$ is

the $m \times p_2$ matrix of coefficients for G, and $U_{m\times n}$ is the $m \times n$ matrix of independent measurement errors. We previously demonstrated that such a decomposition of the variance exists under general conditions typically satisfied in population genomic experiments (Leek and Storey, 2008).

In the training set, the biological classes are known, so S is known and fixed. But the matrices B, Γ and G must be estimated. fSVA first performs surrogate variable analysis (SVA) on the training database in order to identify surrogates for batch effects in the training samples. The training set can be "cleaned" of batch effects by regressing the effect of the surrogate variables out of the data for each feature. Any classification algorithm can then be developed on the basis of the clean training data set.

SVA is an iterative algorithm that alternates between two steps. First SVA estimates the probabilities $\pi_{i\gamma} = \Pr(\gamma_{i\cdot} \neq \vec{0}|X,S,\hat{G}), \pi_{ib} = \Pr(b_{i\cdot} \neq \vec{0}|\gamma_{i\cdot} \neq 0,X,S,\hat{G})$ using an empirical Bayes' estimation procedure (Leek and Storey, 2008; Efron, 2004; Storey *et al.*, 2005). These probabilities are then combined to define an estimate of the probability that a gene is associated with unmeasured confounders, but not with the group outcome

$$\begin{aligned}
\pi_{iw} &= \Pr(b_{i\cdot} = \vec{0} \;\&\; \gamma_{i\cdot} \neq \vec{0}|X,S,\hat{G}) \\
&= \Pr(b_{i\cdot} = \vec{0}|\gamma_{i\cdot} \neq 0,X,S,\hat{G})\Pr(\gamma_{i\cdot} \neq \vec{0}|X,S,\hat{G}) \\
&= (1-\pi_{ib})\pi_{i\gamma}
\end{aligned}$$

The second step of the SVA algorithm weighs each row of the expression ma-

trix X by the corresponding probability weight $\hat{\pi}_{iw}$ and performs a singular value decomposition of the weighted matrix. Letting $\hat{w}_{ii} = \hat{\pi}_{iw}$ the decomposition can be written $\hat{W}X = UDV^T$. After iterating between these two steps, the first p_2 weighted left singular vectors of X are used as estimates of G. An estimate of p_2 can be obtained either through permutation (Buja and Eyuboglu, 1992) or asymptotic (Leek, 2011) approaches.

Once estimates $\hat{G}$ have been obtained, it is possible to fit the regression model in equation 2 using standard least squares. The result are estimates for the coefficients $\hat{B}$ and $\hat{\Gamma}$. Batch effects can be removed from the training set by setting $\hat{X}^{clean} = X - \hat{\Gamma}\hat{G}$. Any standard prediction algorithm can then be applied to $\hat{X}^{clean}$ to develop a classifier based on batch-free genomic data. The result is a prediction function $f(\hat{X}^{clean}_{\cdot j})$ that predicts the outcome variable y_j based on the clean expression matrix.

2.2 Removing batch effects from new samples

Removing batch effects from the training database is accomplished using standard population genomic SVA batch correction. But the application of classifiers to new genomic samples requires batch correction of individual samples when both the batch and outcome variables are unknown. The fSVA algorithm borrows strength from the training database to perform this batch correction.

To remove batch effects from a new sample $X_{\cdot j'}$, it is first appended to the training data to create an augmented expression matrix $X^{j'} = [XX_{\cdot j'}]$ where $[\cdot]$ denotes concatenation of columns. To estimate the values of G for the new sam-

ple, fSVA uses a weighted singular value decomposition, using the probability weights estimated from the training database $\hat{W}X^{j'} = U^{j'}D^{j'}V^{j'T}$. The result is an estimate $\hat{G}^{j'}$ that includes a column for the new sample. Note that only one new sample was appended. Had all the new samples been appended at once, the singular value decomposition would be highly influenced by the similarity in the new samples, rather than detecting similarities between the new sample and the database samples.

To remove batch effects from the new sample, fSVA uses the coefficients estimated from the training database $\hat{\Gamma}$ and the estimated surrogate variables: $\hat{X}^{cleanj'} = X^{j'} - \hat{\Gamma}\hat{G}^{j'}$. If there are n training samples, then the $(n+1)$st column of $X^{cleanj'}$ represents the new clean sample. The classifier built on the clean training data can be applied to this clean data set to classify the new sample.

3 Fast fSVA methodology

fSVA requires that a new singular value decomposition be applied to the augmented expression matrix once for each new sample. Although this is somewhat computationally intensive, in typical personalized medicine applications, sample collection and processing will be spread over a long period of time. In this setting, computational time is not of critical concern. However, for evaluating the fSVA methodology or developing new classifiers using cross-validation, it is important to be able to quickly calculate clean expression values for test samples.

We propose an approximate fSVA algorithm that greatly reduces comput-

ing time by performing a streaming singular value decomposition (Warmuth and Kuzmin, 2007, 2008). The basic idea behind our computation speed-up is to perform the singular value decomposition once on the training data, save the left singular vectors and singular values, and use them to calculate approximate values for the right singular values in new samples.

When removing batch effects from the training data, the last step is a weighted singular value decomposition of the training expression matrix $WX = UDV^T$. After convergence, the first p_2 columns of the matrix V are the surrogate variables for the training set. Since U and V are orthonormal matrices, we can write $V^T = D^{-1}U^TWX$. The matrix $P = D^{-1}U^TW$ projects the columns of X onto the right singular vectors V^T. Pre-multiplying a set of new samples X^{new} by P results in an estimate of the singular values for the new samples: $\hat{V}^{Tnew} = P^T X^{new}$. The surrogate variable estimates for the new samples consist of the first p_2 columns of $\hat{V}^{Tnew}$. We obtain clean data for the new samples using the estimated coefficients from the training set, identical to the calculation for the exact fSVA algorithm: $\hat{X}^{clean,new} = X^{new} - \hat{\Gamma}\hat{G}^{new}$.

Estimates obtained using this approximate algorithm are not identical to those obtained using the exact fSVA algorithm. The projection matrix used in the approximation, P^T, is calculated using only the samples in the training set. However, there is only a one-sample difference between the projection calculated in the training set and the projection that would be obtained with exact fSVA. As the training set size grows, the approximation is closer and closer to the answer that would be obtained from the exact algorithm. For smaller databases, there is less

computational burden in calculating the exact estimates. However, for large training sets, the computational savings can be dramatic, as described in the simulation below.

4 Simulation Results

We performed a simulation to examine the benefit of fSVA in prediction problems. In order to do this, we simulated data using equation 2 under different distributions of each parameter. We also created discrete probability weights $\pi_{i\gamma}$ and π_{ib}, each equal to 1 to indicate batch- or outcome-affected, and 0 to indicate otherwise. We also varied the distribution of these probability weights (Table 1). We crafted these simulations to mimic scenarios with a subtle outcome and a strong batch effect, which is frequently the case in genomic data.

We also specified that both the simulated database and the simulated new samples have two batches and two outcomes. Each outcome was represented in 50% of the samples in both the database and the new samples. Similarly, each batch was represented in 50% of the database and new samples.

In the database, we varied the amount of confounding between batch and outcome from a Pearson's correlation of 0 to a correlation of over 0.90. This mimics common database structures in publicly available repositories.

Since the new samples are simulating a collection of single samples (such as new patients coming to the doctor), the correlation of batch and outcome within the new samples matrix is unimportant. To have a representative amount of new

Parameter Distributions	
Scenario 1	$B \sim N(0,1)$ $\Gamma \sim N(0,3)$ $U \sim N(0,2)$
Scenario 2	$B \sim N(0,1)$ $\Gamma \sim N(0,4)$ $U \sim N(0,3)$
Scenario 3	$B \sim N(0,1)$ $\Gamma \sim N(0,4)$ $U \sim N(0,3)$

Affected Features	
Scenario 1	50% batch-affected 50% outcome-affected 40% affected by both
Scenario 2	50% batch-affected 50% outcome-affected 40% affected by both
Scenario 3	80% batch-affected 80% outcome-affected 50% affected by both

Table 1: **Specifications for the three simulation scenarios used to show the performance of fSVA.** We performed three simulations under slightly different parameterizations to show the effectiveness of fSVA in improving prediction accuracy. Parameters from equation 2 were simulated using the distributions specified in this table. Additionally, the percentage of features in the simulation affected by batch, outcome, or both are as indicated in this table. Results from these simulations can be found in Figure 1.

samples from each combination of batch and outcome, we found it best to simulate the new samples by leaving the batch and outcome uncorrelated. That way, each of the four test-cases of batch and outcome combinations was represented in 25% of the new samples.

We simulated 100 database samples and 100 new samples using the parameters described above. Each sample had 10000 features. As a control, for each iteration in addition to performing fSVA correction, we performed SVA correction on the simulated database alone, and also performed prediction with no batch correction on the simulated database or new samples.

To quantify the effect that fSVA had on prediction, we performed exact fSVA as described above on the simulated database and new samples. We then performed Prediction Analysis of Microarrays (PAM), a commonly used method for classifying microarrays (Tibshirani *et al.*, 2002). The PAM prediction model was built on the SVA-corrected database, and then used to predict the outcomes on the fSVA-corrected new samples. Each simulation was repeated 100 times for robustness.

We found that in general the prediction accuracy measures for different iterations of a simulation varied highly, but the ordinality remained relatively constant. Therefore to display results we randomly selected three graphs from each of the scenarios, using the `sample` function in `R` (Figure 1). Each of the graphs from the 100 iteractions for each scenario can be found on the author's website.

We found that fSVA improved the prediction accuracy in all of our simulations (Figure 1). Interestingly, exact fSVA generally outperformed fast fSVA at

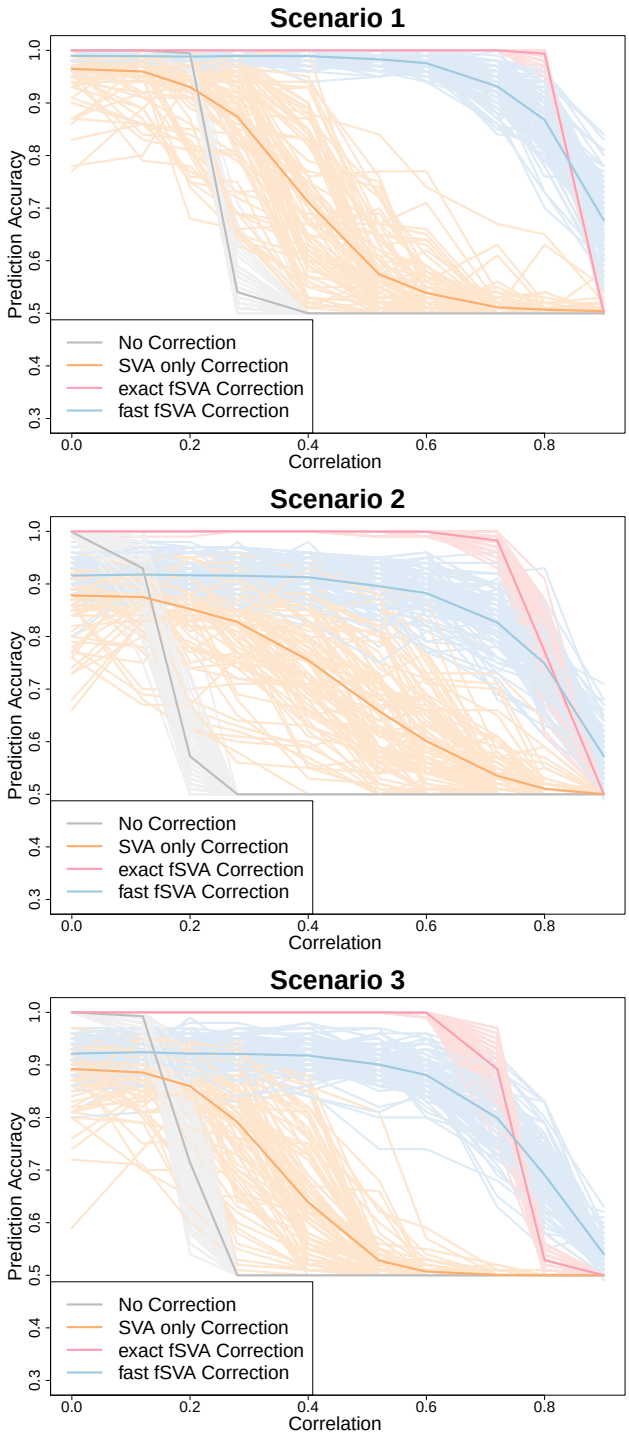

Figure 1: **fSVA improves prediction accuracy of simulated datasets.** We created simulated datasets (consisting of a database and new samples) using model 2 and tested the prediction accuracy of these using R. For each simulated data set we performed either exact fSVA correction, fast fSVA correction, SVA correction on the database only, or no correction. We performed 100 iterations on each simulation scenario described in table 4. These plots show the 100 iterations, as well as the average trend line for each of the four methods investigated.

all of the correlation levels except the highest correlation levels. However both fSVA methods out-performed our control of performing SVA on the database alone. Additionally any method of batch-correction generally outperformed no batch correction whatsoever.

When the batch and outcome were not correlated, we saw ambiguous performance from using fSVA. This is not unexpected since it has been shown that in scenarios with no confounding between batch and outcome, batch has a minimal effect on prediction accuracy (Parker and Leek, 2012). When databases had extreme confounding between batch and outcome (above 0.85) we saw the benefits of all the batch-correction methods drop off. This is because in these situations, SVA on the database cannot differentiate batch and outcome in the database.

While in each of the simulations there was an accuracy cost to using fast fSVA versus exact fSVA, the computational time savings was dramatic. In the scenario described, with 100 samples in the database and 100 new samples, the wall-clock computational time using a standard desktop computer for exact fSVA was 133.9 seconds, versus just 1.3 seconds for fast fSVA. Using 50 samples in the database and 50 new samples, exact fSVA required 17.9 seconds versus 0.4 seconds for fast fSVA. We encourage users to consider both the accuracy and computational times when selecting which algorithm to use for a particular data set.

5 Results from Microarray Studies

We examined the effect that fSVA had on several microarray studies, obtained from the Gene Expression Omnibus (GEO) website (Edgar *et al.*, 2002). All except three of the studies were preprocessed/standardized as described previously. Three of the studies (GSE2034, GSE2603, GSE2990) were obtained from GEO and fRMA-normalized.

Each of the studies was randomly divided into equally-sized "database" and "new sample" subsets. We SVA-corrected the database subset, and then built a predictive model (PAM) on that corrected data. We then performed fSVA correction on the new samples. After performing fSVA correction, we measured the prediction accuracy of the model built on the database by calculating the number of times that the predicted outcome equaled the true outcome status, divided by the number of samples. This process was iterated 100 times for each study to obtain confidence intervals. This method is virtually identical to the simulation described above.

Results from this process can be found below (Table 5). Five of the studies showed significant improvement using fSVA. One study showed marginal improvement, with its 95% confidence interval overlapped zero. Three studies showed a cost to using fSVA, though in all three cases the 95% confidence interval for the true cost overlapped zero.

Study	No Correction	Improvement with fSVA
GSE10927	**0.97** (0.96, 0.97)	**0.02** (0.01, 0.02)
GSE13041	**0.61** (0.59, 0.63)	**0.07** (0.05, 0.10)
GSE13911	**0.93** (0.93, 0.94)	**0.01** (0.00, 0.01)
GSE2034	**0.51** (0.49, 0.52)	**0.03** (0.01, 0.05)
GSE2603	**0.68** (0.66, 0.70)	**-0.02** (-0.04, 0.00)
GSE2990	**0.59** (0.58, 0.61)	**-0.02** (-0.04, 0.00)
GSE4183	**0.89** (0.88, 0.91)	**-0.02** (-0.03, 0.00)
GSE6764	**0.74** (0.72, 0.76)	**0.01** (-0.01, 0.03)
GSE7696	**0.78** (0.76, 0.79)	**0.02** (0.01, 0.04)

Table 2: **fSVA improves prediction accuracy in 5 of the 9 studies examined.** The remaining 4 studies showed indeterminate results since the 95% confidence intervals overlapped zero. In order to find the prediction accuracy results, each of the studies was randomly divided into "database samples" and "new samples". Exact fSVA-correction was then performed as described above. We then built a predictive model (PAM) on the database and tested the prediction accuracy on the new samples.

6 Conclusions

Batch effects have been recognized as a crucial hurdle for population genomics experiments (Leek *et al.*, 2010; Parker and Leek, 2012). They have also been recognized as a critical hurdle in developing genomic signatures for personalized medicine (Micheel *et al.*, 2012). Here we have introduced the first batch correction method specifically developed for prediction problems. Our approach borrows strength from a training set to infer and remove batch effects in individual clinical samples.

We have demonstrated the power of our approach in both simulated and real gene expression microarray data. However, our approach depends on similarity between the training set and the test samples, both in terms of the genes affected and the estimated coefficients. In small training sets, these assumptions may be violated. Similarly, training sets that show near perfect correlation between batch variables and biological classes represent an extreme case that can not be directly corrected using fSVA. An interesting avenue for future research is the use of publicly available microarray data to build increasingly large training databases for batch removal.

The methods we have developed here are available as part of the `sva` Bioconductor package (Leek *et al.*, 2012). Code and data to reproduce this project are available at `https://github.com/hilaryparker/fSVA`.

7 Acknowledgements

This work is partially supported by NIH R01 GM083084 and GM103552, and H.P. was partially funded by the Hopkins Sommer Scholarship.

References

Akey, J. M., Biswas, S., Leek, J. T., and Storey, J. D. (2007). On the design and analysis of gene expression studies in human populations. *Nature Genetics*, **39**(7), 807–8; author reply 808–9.

Baggerly, K. a., Morris, J. S., and Coombes, K. R. (2004). Reproducibility of SELDI-TOF protein patterns in serum: comparing datasets from different experiments. *Bioinformatics (Oxford, England)*, **20**(5), 777–85.

Baggerly, K. a., Coombes, K. R., and Morris, J. S. (2005). Bias, randomization, and ovarian proteomic data: a reply to "producers and consumers". *Cancer informatics*, **1**(2003), 9–14.

Buja, A. and Eyuboglu, N. (1992). Remarks on Parallel Analysis. *Multivariate Behavioral Research*, **27**(4), 509–540.

Chan, I. S. and Ginsburg, G. S. (2011). Personalized medicine: progress and promise. *Annual review of genomics and human genetics*, **12**, 217–44.

Edgar, R., Domrachev, M., and Lash, A. E. (2002). Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Research*, **30**(1), 207–10.

Efron, B. (2004). Large-Scale Simultaneous Hypothesis Testing. *Journal of the American Statistical Association*, **99**(465), 96–104.

Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, **95**(25), 14863–8.

Fare, T. L., Coffey, E. M., Dai, H., He, Y. D., Kessler, D. a., Kilian, K. a., Koch, J. E., LeProust, E., Marton, M. J., Meyer, M. R., Stoughton, R. B., Tokiwa, G. Y., and Wang, Y. (2003). Effects of atmospheric ozone on microarray data quality. *Analytical chemistry*, **75**(17), 4672–5.

Friguet, C., Kloareg, M., and Causeur, D. (2009). A Factor Model Approach to Multiple Testing Under Dependence. *Journal of the American Statistical Association*, **104**(488), 1406–1415.

Gagnon-Bartsch, J. A. and Speed, T. P. (2012). Using control genes to correct for unwanted variation in microarray data. *Biostatistics*, **13**(3), 539—-552.

Johnson, W. E., Li, C., and Rabinovic, A. (2007). Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, **8**(1), 118–27.

Lambert, C. G. and Black, L. J. (2012). Learning from our GWAS mistakes: from experimental design to scientific method. *Biostatistics (Oxford, England)*, **13**(2), 195–203.

Lander, E. S. (1999). Array of hope. *Nature genetics*, **21**(1 Suppl), 3–4.

Leek, J. T. (2011). Asymptotic conditional singular value decomposition for high-dimensional genomic data. *Biometrics*, **67**(2), 344–52.

Leek, J. T. and Storey, J. D. (2007). Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genetics*, **3**(9), 1724–35.

Leek, J. T. and Storey, J. D. (2008). A general framework for multiple testing dependence. *Proceedings of the National Academy of Sciences of the United States of America*, **105**(48), 18718–23.

Leek, J. T., Scharpf, R. B., Bravo, H. C., Simcha, D., Langmead, B., Johnson, W. E., Geman, D., Baggerly, K. A., and Irizarry, R. A. (2010). Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Reviews Genetics*, **11**(10), 733–9.

Leek, J. T., Johnson, W. E., Parker, H. S., Jaffe, A. E., and Storey, J. D. (2012). The sva package for removing batch effects and other unwanted variation in high-throughput experiments. *Bioinformatics*, **28**(6), 882—-883.

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S., Tillinghast, G., Liu, G., Zhou, Y., Herman, D., Li, Y., Deng, Y., Fang, H., Bushel, P., Woods, M., and Zhang, J. (2010). A comparison of batch effect removal methods for enhancement of prediction performance using MAQC-II microarray gene expression data. *The Pharmacogenomics Journal*, **10**(4), 278–91.

Micheel, C. M., Nass, S. J., and Omenn, G. S. (2012). *Evolution of Translational Omics: Lessons Learned and the Path Forward*. The National Academies Press.

Parker, H. S. and Leek, J. T. (2012). The practical effect of batch on genomic prediction. *Statistical Applications in Genetics and Molecular Biology*, **11**(3).

Scharpf, R. B., Ruczinski, I., Carvalho, B., Doan, B., Chakravarti, A., and Irizarry, R. a. (2011). A multilevel model to address batch effects in copy number estimation using SNP arrays. *Biostatistics (Oxford, England)*, **12**(1), 33–50.

Sebastiani, P., Solovieff, N., Puca, A., Hartley, S. W., Melista, E., Dworkis, D. A., Wilk, J. B., Myers, R. H., Steinberg, M. H., Baldwin, C. T., and Perls, T. T. (2010). Genetic Signatures of Exceptional Longevity in Humans. *Science*, **10**, 1126.

Spielman, R. S., Bastone, L. a., Burdick, J. T., Morley, M., Ewens, W. J., and Cheung, V. G. (2007). Common genetic variants account for differences in gene expression among ethnic groups. *Nature genetics*, **39**(2), 226–31.

Storey, J. D., Akey, J. M., and Kruglyak, L. (2005). Multiple locus linkage analysis of genomewide expression in yeast. *PLoS biology*, **3**(8), e267.

Tibshirani, R., Hastie, T., Narasimhan, B., and Chu, G. (2002). Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Sciences*, **99**(10), 6567–72.

Walker, W. L., Liao, I. H., Gilbert, D. L., Wong, B., Pollard, K. S., McCulloch, C. E., Lit, L., and Sharp, F. R. (2008). Empirical Bayes accomodation of batch-effects in microarray data using identical replicate reference samples: application to RNA expression profiling of blood from Duchenne muscular dystrophy patients. *BMC genomics*, **9**, 494.

Warmuth, M. K. and Kuzmin, D. (2007). Randomized PCA Algorithms with Regret Bounds that are Logarithmic in the Dimension. *Advances in Neural Information Processing Systems*, **19**, 1481.

Warmuth, M. K. and Kuzmin, D. (2008). Randomized Online PCA Algorithms with Regret Bounds that are Logarithmic in the Dimension. **9**, 2287–2320.