RUV for normalization of expression array data

Laurent Jacob

February 26, 2016

Abstract

When dealing with large scale gene expression studies, observations are commonly contaminated by sources of unwanted variation such as platforms or batches. Not taking this unwanted variation into account when analyzing the data can lead to spurious associations and to missing important signals. When the analysis is unsupervised, *e.g.* when the goal is to cluster the samples or to build a corrected version of the dataset — as opposed to the study of an observed factor of interest — taking unwanted variation into account can become a difficult task. The factors driving unwanted variation may be correlated with the unobserved factor of interest, so that correcting for the former can remove the latter if not done carefully. *RUVnormalize* implements methods described in Jacob et al. [2012] to estimate and remove unwanted variation from microarray gene expression data. These methods rely on negative control genes and replicate samples.

1 Introduction

Over the last few years, microarray-based gene expression studies involving a large number of samples have been conducted [Cardoso et al., 2007, Cancer Genome Atlas Research Network, 2008], with the goal of helping understand or predict some particular *factors of interest* like the prognosis or the subtypes of a cancer. Such large gene expression studies are often carried out over several years, may involve several hospitals or research centers and typically contain some *unwanted variation*. Sources of unwanted variation can be technical elements such as batches, different platforms or laboratories, or any biological signal which is not the factor of interest of the study such as heterogeneity in ages or different ethnic groups.

Unwanted variation can easily lead to spurious associations. For example when one is looking for genes which are differentially expressed between two subtypes of cancer, the observed differential expression of some genes could actually be caused by differences between laboratories if laboratories are partially confounded with subtypes. When doing clustering to identify new subgroups of the disease, one may actually identify some of the unwanted factors if their effects on gene expression are stronger than the subgroup effect.

If one is interested in predicting prognosis, one may actually end up predicting whether the sample was collected at the beginning or at the end of the study because better prognosis patients were accepted at the end of the study. In this case, the classifier obtained would have little value for predicting the prognosis of new patients.

Similar problems arise when trying to combine several smaller studies rather than working on one large heterogeneous study: in a dataset resulting from the merging of several studies the strongest effect one can observe is generally related to the membership of samples to different studies. A very important objective is therefore to remove this unwanted variation without losing the variation of interest.

A large number of methods have been proposed to tackle this problem, mostly using linear models. When both the factor of interest and the unwanted factors are observed, the problem essentially boils down to a linear regression [Johnson et al., 2007]. When the factor of interest is observed but the unwanted factors are not, the latter need to be estimated before a regression is possible. This is typically done using the covariance structure of the gene expression matrix [Kang et al., 2008], the residuals after an ordinary regression [Leek and Storey, 2007, Listgarten et al., 2010] or negative control genes [Gagnon-Bartsch and Speed, 2012]. Finally if the factor of interest itself is not defined, some methods [Alter et al., 2000] use singular value decomposition (SVD) on gene expression to identify and remove the unwanted variation and others [Benito et al., 2004] remove observed batches by linear regression.

RUVnormalize addresses this latter case where there is no predefined factor of interest. This situation arises when performing unsupervised estimation tasks such as clustering or PCA, in the presence of unwanted variation. It can also be the case that one needs to normalize a dataset without knowing which factors of interest will be studied. Our main objective is to correct the gene expression by estimating and removing the unwanted variation, without removing the — unobserved — variation of interest.

For more detail about the statistical model and method, see Jacob et al. [2012] and references therein.

2 Software features

RUVnormalize takes as input gene expression data, negative control genes and replicate samples, and offers the following functionalities:

Gene expression correction *RUVnormalize* estimates the unwanted variation from negative control genes or replicate samples and removes it from the input gene expression data, returning a corrected matrix.

Representation *RUVnormalize* provides a function to represent the influence of unwanted variation on gene expression.

3 Case studies

We now show on a particular dataset how *RUVnormalize* can be used to remove unwanted variation from gene expression data.

Vawter et al. [2004] systematically measured the expression of 12,600 genes for 5 male and 5 female patients, with the goal to study gender related differential expression. The samples come from different brain regions and are hybridized from different labs, both of which affect gene expression. Ideally, a correction method applied to this dataset would remove the effect of these unwanted sources of variation without affecting the gender signal.

We apply various correction methods on this dataset and assess how well the corrected data clusters by gender.

3.1 Loading the library and the data

We load the *RUVnormalize* package by typing or pasting the following codes in R command line. We also need the *spams* package.

```
> library(RUVnormalize)
> library(RUVnormalizeData)
```

We then load the expression data, control genes and known factors affecting the expression:

```
> data('gender', package='RUVnormalizeData')
> Y <- t(exprs(gender))
> X <- as.numeric(phenoData(gender)$gender == 'M')
> X <- X - mean(X)
> X <- cbind(X/(sqrt(sum(X^2))))
> chip <- annotation(gender)
> ## Extract regions and labs for plotting purposes
> lregions <- sapply(rownames(Y),FUN=function(s) strsplit(s,'_')[[1]][2])
> llabs <- sapply(rownames(Y),FUN=function(s) strsplit(s,'_')[[1]][3])
> ## Dimension of the factors
> m <- nrow(Y)
> n <- ncol(Y)
> p <- ncol(X)
> Y <- scale(Y, scale=FALSE) # Center gene expressions
> cIdx <- which(featureData(gender)$isNegativeControl) # Negative control genes
> ## Number of genes kept for clustering, based on their variance
> nKeep <- 1260
```

We prepare variables which will then be used to plot the data before and after correction.

```
> ## Prepare plots
> annot <- cbind(as.character(sign(X)))
> colnames(annot) <- 'gender'
> plAnnots <- list('gender'='categorical')
> lab.and.region <- apply(rbind(lregions, llabs),2,FUN=function(v) paste(v,collapse='_'))
> gender.col <- c('-1' = "deeppink3", '1' = "blue")
```

Gene expression in this dataset is strongly affected by a platform effect. This effect is reasonably well corrected by centering the data by platform, so we apply this centering as a pre-processing.

```
> ## Remove platform effect by centering.
> Y[chip=='hgu95a.db',] <- scale(Y[chip=='hgu95a.db',], scale=FALSE)
> Y[chip=='hgu95av2.db',] <- scale(Y[chip=='hgu95av2.db',], scale=FALSE)
```

Some correction methods use a table describing which samples are replicates of each others. The table has as many columns as the largest set of replicates for one sample. Each row corresponds to a set of replicates of the same sample and gives the row indices of the replicates in the gene expression matrix, padded with -1 entries.

```
> ## Prepare control samples
> scIdx <- matrix(-1,84,3)
> rny <- rownames(Y)
> added <- c()
> c <- 0
> # Replicates by lab
> for(r in 1:(length(rny) - 1)){
+   if(r %in% added)
+     next
+   c <- c+1
+   scIdx[c,1] <- r
+   cc <- 2
+   for(rr in seq(along=rny[(r+1):length(rny)])){
+     if(all(strsplit(rny[r],'_')[[1]][-3] ==  strsplit(rny[r+rr],'_')[[1]][-3])){
+       scIdx[c,cc] <- r+rr
+       cc <- cc+1
+       added <- c(added,r+rr)
+     }
+   }
+ }
> scIdxLab <- scIdx
> scIdx <- matrix(-1,84,3)
> rny <- rownames(Y)
> added <- c()
> c <- 0
> ## Replicates by region
> for(r in 1:(length(rny) - 1)){
+   if(r %in% added)
+     next
+   c <- c+1
+   scIdx[c,1] <- r
+   cc <- 2
+   for(rr in seq(along=rny[(r+1):length(rny)])){
+     if(all(strsplit(rny[r],'_')[[1]][-2] ==  strsplit(rny[r+rr],'_')[[1]][-2])){
+       scIdx[c,cc] <- r+rr
+       cc <- cc+1
+       added <- c(added,r+rr)
+     }
+   }
+ }
> scIdx <- rbind(scIdxLab,scIdx)
```

3.2 Correction

We now apply the correction methods and plot the corrected data. More specifically after each correction, we apply k-means clustering to the corrected gene expression matrix, and plot the projection of the samples onto the space spanned by the first two principal components. We plot the projections as we go, and summarize the clustering qualities in a table at the end of the vignette. We use the function `clScore` to compare the partition of the samples obtained using k-means to the ground truth (partition by gender).

As described in Jacob et al. [2012], we only keep the 10% genes with the largest variance for clustering an computing the principal components.

As a baseline, we start with the uncorrected gene expression matrix:

```
> ## Sort genes by their standard deviation
> sdY <- apply(Y, 2, sd)
> ssd <- sort(sdY, decreasing=TRUE, index.return=TRUE)$ix
> ## Cluster the samples
> kmres <- kmeans(Y[, ssd[1:nKeep], drop=FALSE], centers=2, nstart=200)
> vclust <- kmres$cluster
> ## Compute the distance between clustering by gender
> ## and clustering obtained by k-means
> uScore <- clScore(vclust,X)
```

We then plot the first two principal components for the uncorrected gene expression matrix.

The plot suggests that without correction, the observed gene expression is mainly driven by a lab effect (PC1) and a brain region effect (PC2).

In the rest of the vignette, we apply the same steps (clustering and PCA plot) after centering genes by lab-region batch, using naive RUV-2, random naive RUV-2, the replicate based correction and iterative corrections based on replicates and negative control genes only. See Jacob et al. [2012] for more details about the correction methods.

```
> ## Centering by region-lab
> YmeanCorr <- Y
> for(rr in unique(lregions)){
+   for(ll in unique(llabs)){
+     YmeanCorr[(lregions==rr)&(llabs==ll),] <- scale(YmeanCorr[(lregions==rr)&(llabs==ll),
+   }
+ }
> sdY <- apply(YmeanCorr, 2, sd)
> ssd <- sort(sdY,decreasing=TRUE,index.return=TRUE)$ix
> kmresMC <- kmeans(YmeanCorr[,ssd[1:nKeep],drop=FALSE],centers=2,nstart=200)
> vclustMC <- kmresMC$cluster
> MCScore <- clScore(vclustMC, X)
```

The plot shows that centering removed the lab and brain region effects, but not in a way that leads to a clustering by gender. The following methods lead to a removal of the lab and brain region effects which leads to a better clustering of the samples by gender.

```
> ## Naive RUV-2 no shrinkage
> k <- 20
> nu <- 0
> nsY <- naiveRandRUV(Y, cIdx, nuCoeff=0, k=k)
> sdY <- apply(nsY, 2, sd)
> ssd <- sort(sdY,decreasing=TRUE,index.return=TRUE)$ix
> kmres2ns <- kmeans(nsY[,ssd[1:nKeep],drop=FALSE],centers=2,nstart=200)
> vclust2ns <- kmres2ns$cluster
> nsScore <- clScore(vclust2ns, X)
```

```
> ## Naive RUV-2 + shrinkage
>
> k <- m
> nu.coeff <- 1e-3
> nY <- naiveRandRUV(Y, cIdx, nuCoeff=nu.coeff, k=k)
> svdResUncorr <- svdPlot(Y[, ssd[1:nKeep], drop=FALSE],
+                         annot=annot,
+                         labels=lab.and.region,
+                         svdRes=NULL,
+                         plAnnots=plAnnots,
+                         kColors=gender.col, file=NULL)
```

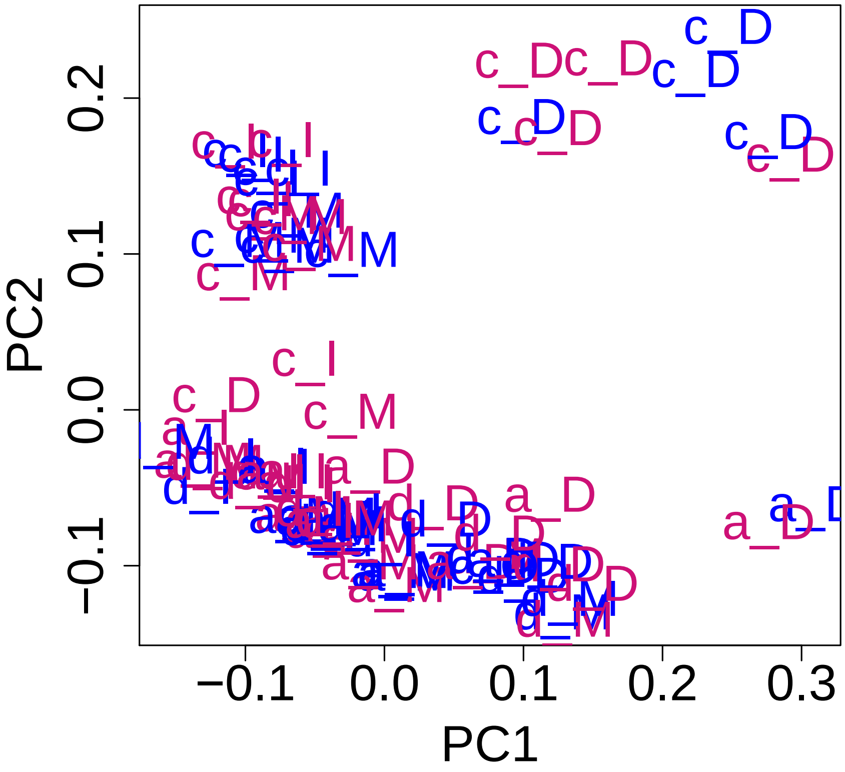

Figure 1: Samples of the gender study represented in the space of their first two principal components before correction. Blue samples are males, pink samples are females. The upper case letter represents the lab, the lower case one is the brain region.

```
> svdResMC <- svdPlot(YmeanCorr[, ssd[1:nKeep], drop=FALSE],
+                     annot=annot,
+                     labels=lab.and.region,
+                     svdRes=NULL,
+                     plAnnots=plAnnots,
+                     kColors=gender.col, file=NULL)
```

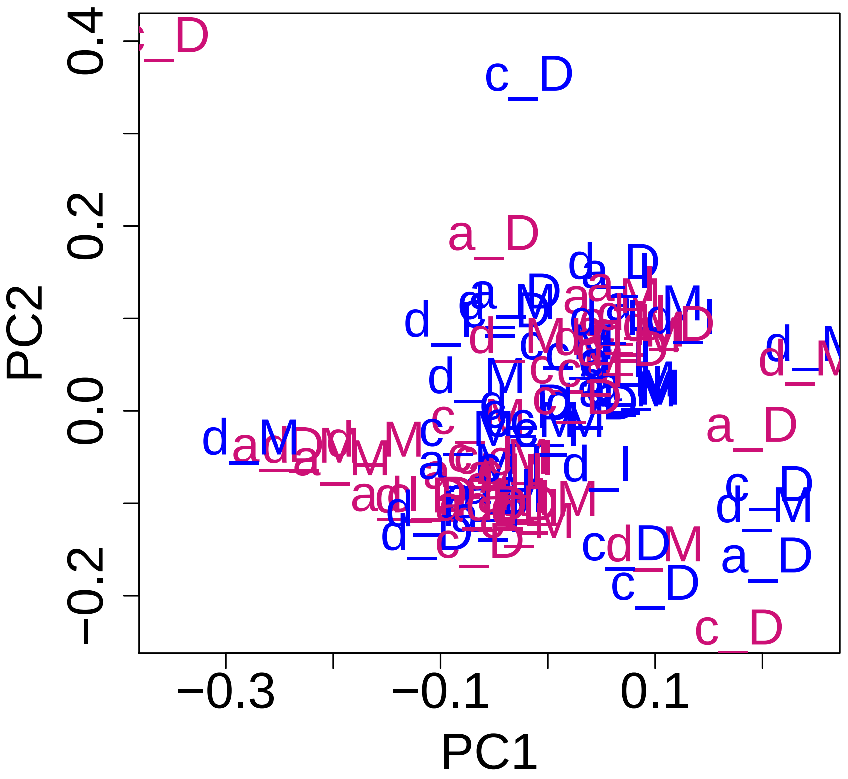

Figure 2: Samples of the gender study represented in the space of their first two principal components after mean centering genes withing each region/lab groups. Blue samples are males, pink samples are females. The upper case letter represents the lab, the lower case one is the brain region.

```
> svdRes2ns <- svdPlot(nsY[, ssd[1:nKeep], drop=FALSE],
+                      annot=annot,
+                      labels=lab.and.region,
+                      svdRes=NULL,
+                      plAnnots=plAnnots,
+                      kColors=gender.col, file=NULL)
```

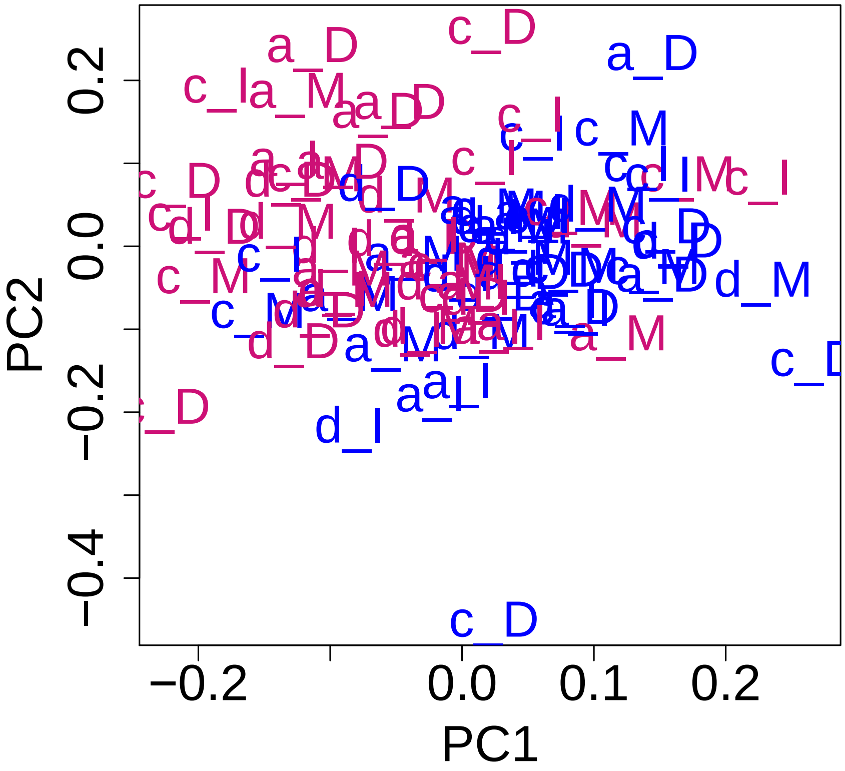

Figure 3: Samples of the gender study represented in the space of their first two principal components after applying the naive RUV-2 correction `naiveRandRUV` with rank reduction ($k = 20$) and no shrinkage ($\nu = 0$). Blue samples are males, pink samples are females. The upper case letter represents the lab, the lower case one is the brain region.

```
> sdY <- apply(nY, 2, sd)
> ssd <- sort(sdY,decreasing=TRUE,index.return=TRUE)$ix
> kmres2 <- kmeans(nY[,ssd[1:nKeep],drop=FALSE],centers=2,nstart=200)
> vclust2 <- kmres2$cluster
> nScore <- clScore(vclust2,X)
```

```
> ## Replicate-based
>
> sRes <- naiveReplicateRUV(Y, cIdx, scIdx, k=20)
> sdY <- apply(sRes$cY, 2, sd)
> ssd <- sort(sdY,decreasing=TRUE,index.return=TRUE)$ix
> kmresRep <- kmeans(sRes$cY[,ssd[1:nKeep],drop=FALSE],centers=2,nstart=200)
> vclustRep <- kmresRep$cluster
> RepScore <- clScore(vclustRep,X)
```

The last two correction methods are iterative: they start by a computing a naive estimate of the $W\alpha$ unwanted variation term, then estimate a term of interest $X\beta$ from the residuals $Y - W\alpha$, re-estimate $W\alpha$ from $Y - X\beta$ and iterate between these two steps for a fixed number of steps or until some convergence is reached.

In these example, the estimation of $X\beta$ given $W\alpha$ is done using a sparse dictionary learning method [Mairal et al., 2010]. The choice of the regularization parameters is discussed in Jacob et al. [2012]. The `paramXb` variable corresponds to the parameters of the sparse dictionary learning method. The `D`, `batch`, `iter` and `mode` should not be modified unless you are familiar with Mairal et al. [2010] and know precisely what you are doing. `K` corresponds to the rank of X, *i.e.*, p in our notation, and `lambda` is the regularization parameter. Large values of `lambda` lead to sparser, more shrunk estimates of β.

```
> if (require(spams)){
+   ## Iterative replicate-based
+   cEps <- 1e-6
+   maxIter <- 30
+   p <- 20
+
+   paramXb <- list()
+   paramXb$K <- p
+   paramXb$D <- matrix(c(0.),nrow = 0,ncol=0)
+   paramXb$batch <- TRUE
+   paramXb$iter <- 1
+
+   ## l1
+   paramXb$mode <- 'PENALTY'
```

```
> svdRes2 <- svdPlot(nY[, ssd[1:nKeep], drop=FALSE],
+                    annot=annot,
+                    labels=lab.and.region,
+                    svdRes=NULL,
+                    plAnnots=plAnnots,
+                    kColors=gender.col, file=NULL)
```

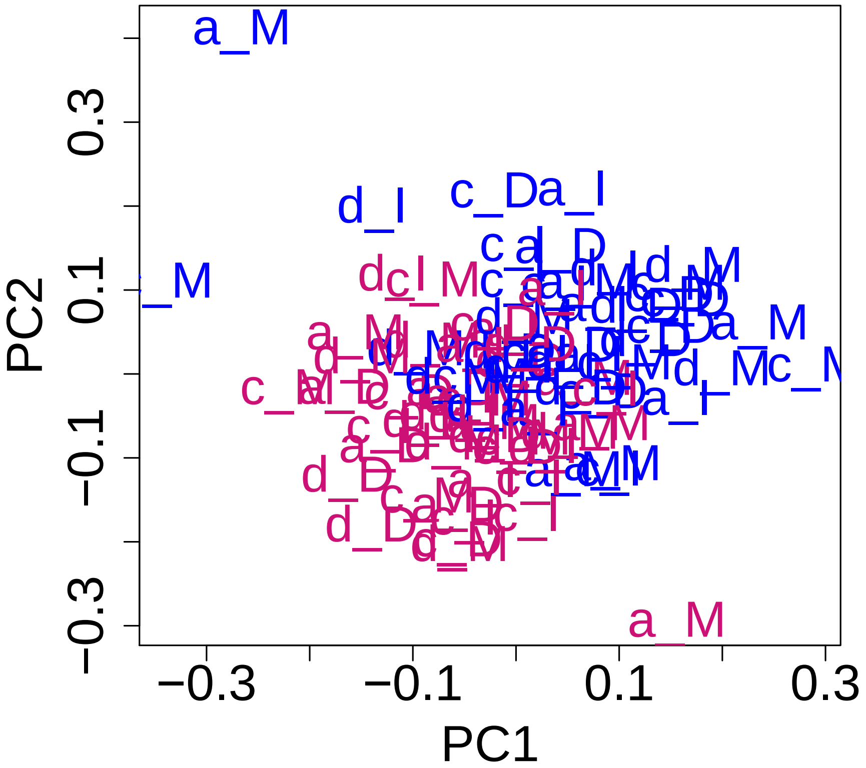

Figure 4: Samples of the gender study represented in the space of their first two principal components after applying the naive RUV-2 correction `naiveRandRUV` using no rank reduction ($k = m$) but shrinkage $\nu \neq 0$. Blue samples are males, pink samples are females. The upper case letter represents the lab, the lower case one is the brain region.

```
> svdResRep <- svdPlot(sRes$cY[, ssd[1:nKeep], drop=FALSE],
+                       annot=annot,
+                       labels=lab.and.region,
+                       svdRes=NULL,
+                       plAnnots=plAnnots,
+                       kColors=gender.col, file=NULL)
```

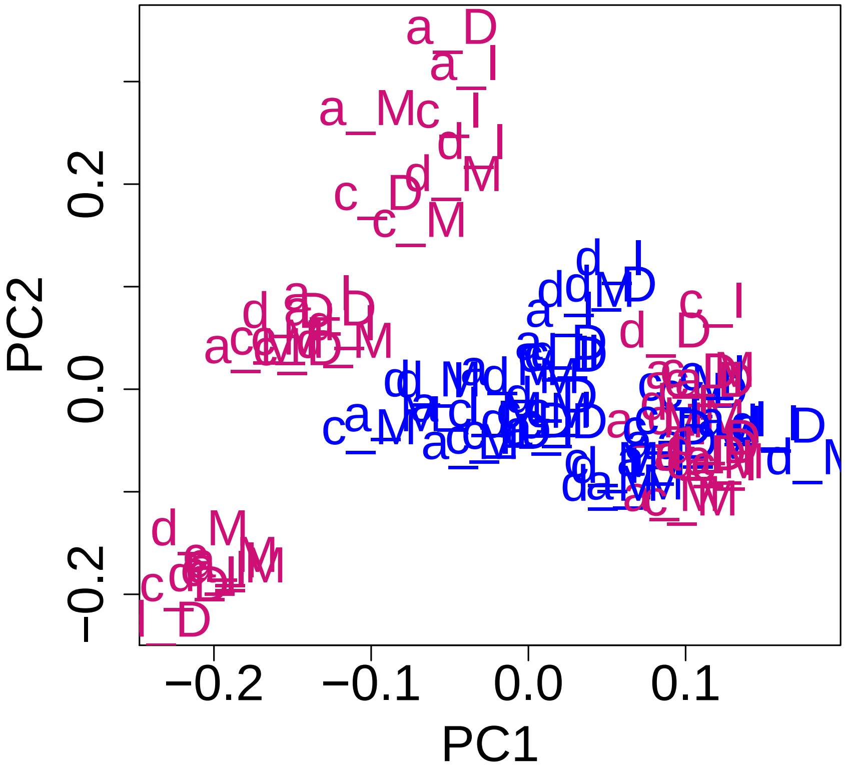

Figure 5: Samples of the gender study represented in the space of their first two principal components after applying the replicate based correction `naiveReplicateRUV`. Blue samples are males, pink samples are females. The upper case letter represents the lab, the lower case one is the brain region.

```
+   paramXb$lambda <- 0.25
+
+   iRes <- iterativeRUV(Y, cIdx, scIdx, paramXb, k=20, nu.coeff=0,
+   cEps, maxIter,
+   Wmethod='rep', wUpdate=11)
+
+   ucY <- iRes$cY
+
+   sdY <- apply(ucY, 2, sd)
+   ssd <- sort(sdY,decreasing=TRUE,index.return=TRUE)$ix
+
+   kmresIter <- kmeans(ucY[,ssd[1:nKeep]],centers=2,nstart=200)
+   vclustIter <- kmresIter$cluster
+   IterScore <- clScore(vclustIter,X)
+ }else{
+     IterScore <- NA
+ }

> if (require(spams)){
+   ## Iterated ridge
+   paramXb <- list()
+   paramXb$K <- p
+   paramXb$D <- matrix(c(0.),nrow = 0,ncol=0)
+   paramXb$batch <- TRUE
+   paramXb$iter <- 1
+   paramXb$mode <- 'PENALTY' #2
+   paramXb$lambda <- 6e-2
+   paramXb$lambda2 <- 0
+
+   iRes <- iterativeRUV(Y, cIdx, scIdx=NULL, paramXb, k=nrow(Y), nu.coeff=1e-3/2,
+   cEps, maxIter,
+   Wmethod='svd', wUpdate=11)
+
+   nrcY <- iRes$cY
+
+   sdY <- apply(nrcY, 2, sd)
+   ssd <- sort(sdY,decreasing=TRUE,index.return=TRUE)$ix
+
+   kmresIter <- kmeans(nrcY[,ssd[1:nKeep]],centers=2,nstart=200)
+   vclustIter <- kmresIter$cluster
+   IterRandScore <- clScore(vclustIter,X)
+ }else{
+     IterRandScore <- NA
+ }
```

Finally, we summarize the clustering errors obtained after each correction in a single table:

```
> scores <- c(uScore, MCScore, nsScore, nScore, RepScore, IterScore, IterRandScore)
> names(scores) <- c('Uncorrected', 'Centered', 'Naive RUV-2', 'Naive + shrink', 'Replicate
> print('Clustering errors after each correction')

[1] "Clustering errors after each correction"

> print(scores)

      Uncorrected          Centered        Naive RUV-2    Naive + shrink
        0.9997457         0.9725210          0.7507730         0.6737471
       Replicates Replicates + iter  Shrinkage + iter
        0.7702779                NA                NA
```

4 Session Information

```
R version 3.2.3 (2015-12-10)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 14.04.4 LTS

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

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

other attached packages:
[1] RUVnormalizeData_0.104.0 Biobase_2.30.0           BiocGenerics_0.16.1
[4] RUVnormalize_1.4.1
```

```
loaded via a namespace (and not attached):
[1] tools_3.2.3
```

References

O. Alter, P. O. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proc Natl Acad Sci U S A*, 97(18):10101–10106, Aug 2000.

Monica Benito, Joel Parker, Quan Du, Junyuan Wu, Dong Xiang, Charles M Perou, and J. S. Marron. Adjustment of systematic microarray data biases. *Bioinformatics*, 20(1): 105–14, Jan 2004.

Cancer Genome Atlas Research Network. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature*, 455(7216):1061–1068, Oct 2008. doi: 10.1038/nature07385. URL `http://dx.doi.org/10.1038/nature07385`.

Fatima Cardoso, Martine Piccart-Gebhart, Laura Van't Veer, Emiel Rutgers, and TRANSBIG Consortium. The MINDACT trial: the first prospective clinical validation of a genomic tool. *Molecular oncology*, 1(3):246–251, Dec 2007. ISSN 1878-0261. doi: 10.1016/j.molonc.2007.10.004. URL `http://dx.doi.org/10.1016/j.molonc.2007.10.004`.

Johann A. Gagnon-Bartsch and Terence P. Speed. Using control genes to correct for unwanted variation in microarray data. *Biostatistics*, 13(3):539–552, Jul 2012. ISSN 1468-4357. doi: 10.1093/biostatistics/kxr034. URL `http://dx.doi.org/10.1093/biostatistics/kxr034`.

L. Jacob, J. Gagnon-Bartsch, and T. P. Speed. Correcting gene expression data when neither the unwanted variation nor the factor of interest are observed. Technical report, arXiv, 2012. URL `http://arxiv.org/abs/1211.4259`.

W. Evan Johnson, Cheng Li, Department Biostatistics, Computational Biology, and Ariel Rabinovic. Adjusting batch effects in microarray expression data using empirical bayes methods. *Biostatistics*, 1(8):118–127, 2007.

Hyun Min Kang, Chun Ye, and Eleazar Eskin. Accurate discovery of expression quantitative trait loci under confounding from spurious and genuine regulatory hotspots. *Genetics*, 180(4):1909–1925, Dec 2008. doi: 10.1534/genetics.108.094201. URL `http://dx.doi.org/10.1534/genetics.108.094201`.

Jeffrey T Leek and John D Storey. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genet*, 3(9):1724–1735, Sep 2007. doi: 10.1371/journal.pgen.0030161. URL `http://dx.doi.org/10.1371/journal.pgen.0030161`.

Jennifer Listgarten, Carl Kadie, Eric E Schadt, and David Heckerman. Correction for hidden confounders in the genetic analysis of gene expression. *Proc Natl Acad Sci U S A*, 107(38):16465–16470, Sep 2010. doi: 10.1073/pnas.1002425107. URL `http://dx.doi.org/10.1073/pnas.1002425107`.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

Marquis P. Vawter, Simon Evans, Prabhakara Choudary, Hiroaki Tomita, Jim Meador-Woodruff, Margherita Molnar, Jun Li, Juan F. Lopez, Rick Myers, David Cox, Stanley J. Watson, Huda Akil, Edward G. Jones, and William E. Bunney. Gender-specific gene expression in post-mortem human brain: localization to sex chromosomes. *Neuropsychopharmacology*, 29(2):373–384, Feb 2004. ISSN 0893-133X. doi: 10.1038/sj.npp.1300337. URL `http://dx.doi.org/10.1038/sj.npp.1300337`.