Practical Batch-Effects

Maarten van Iterson

July 4, 2017

Contents

Introduction

In this practical you will learn how to perform batch correction on RNAseq data using the packages RUVSeq and sva. The example data that will be used is a subset of the HapMap RNAseq data described by Pickrell and Montgomery (Pickrell et al. 2010, Montgomery et al. (2010)). A combined dataset containing the RNAseq data of both papers is available from the ReCount website (Frazee, Langmead, and Leek 2011). As phenotype information we have Population (CEU/YRI) and Gender (Male/Female). In some of the Exercises we will assume the population origin of the samples is a unknown batch effect.

The practical consists of four parts:

- download the data and some preprocessing
- find diff. expr. genes between Male/Female using limma's `voom` or edgeR with population as a known batch in the design matrix
- estimate batch effects using RUVSeq and find diff. expr. genes
- estimate batch effects using sva and find diff. expr. genes

Get the data:

Run the following two code chunks to get the data in your R-environment.

```
library(Biobase)
monpick <- "http://bowtie-bio.sourceforge.net/recount/ExpressionSets/montpick_eset.RData"
load(url(monpick))
head(pData(montpick.eset))
counts <- exprs(montpick.eset)
counts[1:5, 1:5]
```

Unfortunately, gender information is not provided. This can be obtained from the 1000genomes website: ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/technical/working/20130606_sample_info/20130606_sample_info.xlsx Download the sample information file and save the sheet with sample info as csv-file and preprocess like this:

```
sample.info.file <- "/20130606_sample_info.csv"
sample.info <- read.table(, header=TRUE, sep="\t")
head(sample.info)
pdata <- merge(pData(montpick.eset), sample.info, by.x="sample.id", by.y="Sample", all.x=TRUE)
pdata[is.na(pdata$Gender),]
counts <- counts[,!is.na(pdata$Gender)]
pdata <- pdata[!is.na(pdata$Gender),]
pdata <- droplevels(pdata)
dim(pdata)
dim(counts)
```

We had to remove two samples because for these no phenotype information was available. Furthermore, we dropped the additional factor levels as these will interfere later with the creation of design-matrices.

Now we can start our analyzes!

Finding diff. expr. genes using `limma` or `edgeR`

First we will find diff. expr. genes using either limma's `voom` or edgeR which one is up to you!

Preprocessing and data inspection

Exercise 1: Construct a *egdeR* `DGEList` with a group-variable the Gender information. Optionally, remove low expressed genes.

Solution 1:

```
isexpr <- rowSums(counts) > 50
counts <- counts[isexpr,]
dim(counts)
library(edgeR)
d <- DGEList(counts, group=pdata$Gender)
d <- calcNormFactors(d)
```

Exercise 2: Inspect the data using a Multi-dimensional scaling plot to see potential batches in the data. Use coloring and labels to see which, gender or population, has the strongest effect.

Solution 2:

```r
Gender <- substring(pdata$Gender,1,1)
colGen <- ifelse(Gender=="m","blue","red")
plotMDS(d, labels = Gender,top = 50, col=colGen, gene.selection="common", prior.count = 5)
```

```r
Population <- substring(pdata$population,1,1)
colPop <- ifelse(Population=="C","blue","red")
plotMDS(d, labels = Gender,top = 50, col=colPop, gene.selection="common", prior.count = 5)
```

Fitting using `voom`

Exercise 3a: Fit a linear model correcting for the population structure using `voom`.

Solution 3a:

```r
design <- model.matrix(~Gender + Population, data = pdata)
v <- voom(d, design, plot = TRUE)
```

```r
fit <- lmFit(v, design)
fit <- eBayes(fit)
```

```r
summary(decideTests(fit))
topTable(fit)
```

Exercise 3b: Fit a linear model correcting for the population structure using *edgeR* (you can reuse the `DGEList`).

Solution 3b:

```r
d1 <- estimateGLMCommonDisp(d, design)
d1 <- estimateGLMTagwiseDisp(d1, design)
```

```r
fit <- glmFit(d1, design)
lrt <- glmLRT(fit, coef=2)
topTags(lrt)
```

Optional Exercise: Annotate the top genes using org.Hs.eg.db. Are these the genes you would had expected?

Batch effect correction using *RUVseq*

Now we will assume the population of the samples was unknown and investigate if we can correct for this using the method implemented in *RUVseq*. Since, we do not have negative controls we will use a set of empirical controls. Empirical controls are just the genes that do not show diff. expr. for the phenotype of interest.

Exercise 4: Find a set of empirical control genes.

Solution 4:

```
top <- topTags(lrt, n=Inf)$table
empirical <- rownames(d)[which(!(rownames(d) %in% rownames(top)[1:5000]))]
```

Exercise 5: Run `RUVg` and inspect the effect on the data using the plotRLE and/or plotPCA from *RUVSeq*.

Solution 5:

```
library(RUVSeq)
rownames(pdata) <- pdata$sample.id
set <- newSeqExpressionSet(d$counts, phenoData = pdata$Gender)
set
```

```
corrected <- RUVg(set, empirical, k=1)
```

```
op <- par(mfcol=c(2, 1))
plotRLE(set, outline=FALSE, ylim=c(-2, 2), col=colPop, las=2)
plotRLE(corrected, outline=FALSE, ylim=c(-2, 2), col=colGen, las=2)
par(op)
```

```
op <- par(mfcol=c(2, 1))
plotPCA(set, col=colPop, cex=1.2)
plotPCA(corrected, col=colGen, cex=1.2)
par(op)
```

Exercise 6: Can you think of a way to see what the estimated unwanted variation represents? What does it represent?

Solution 6:

```
ruv1 <- corrected$W_1
plot(ruv1, pdata$Population)
```

Exercise 7: Fit a linear model correcting for the factor of unwanted variation. You can reuse the *DGEList* and again it is up to you to use `voom` or *edgeR*.

Solution 7:

```
designruv <- model.matrix(~Gender, data = pdata)
designruv <- cbind(designruv, ruv1)
d2 <- estimateGLMCommonDisp(d, designruv)
d2 <- estimateGLMTagwiseDisp(d2, designruv)
fit <- glmFit(d2, designruv)
lrt <- glmLRT(fit, coef=2)
topTags(lrt)
```

Batch effect correction using *sva*

Now we will use the *sva*-package to estimate the unwanted variation introduced by the different populations.

> Exercise 8: What are the optimal number of surrogate variables to we should correct for?

Solution 8:

```r
suppressPackageStartupMessages(library(sva))
designsva <- model.matrix(~Gender, data = pdata)
n.sv <- num.sv(d$counts, designsva, method="leek")
n.sv
```

If find this number too high and we can not even correct for all these; you will get a error-message if you try. I suppose we use just one!

> Exercise 9: To estimate the surrogate variable we need to define our null hypothesis. What is our null hypothesis? Run `svaseq`.

Solution 9:

```r
designsva0 <- model.matrix(~1, data = pdata)
svseq <- svaseq(d$counts, designsva, designsva0, n.sv=1)
```

> Exercise 10: Can you think of a way to see what the estimated unwanted variation represents? What does it represent?

Solution 10:

```r
plot(svseq$sv, pdata$Population)
```

> Exercise 11: Fit a linear model correcting for the surrogate variable. You can reuse the `DGEList` and again it is up to you to use `voom` or *edgeR*.

Solution 11:

```r
designsva <- cbind(designsva, svseq$sv)
v <- voom(d, designsva, plot = TRUE)
fit <- lmFit(v, designsva)
fit <- eBayes(fit)
summary(decideTests(fit))
topTable(fit)
```

References

Frazee, A. C., B. Langmead, and J. T. Leek. 2011. "ReCount: a multi-experiment resource of analysis-ready RNA-seq gene count datasets." *BMC Bioinformatics* 12: 449.

Montgomery, S. B., M. Sammeth, M. Gutierrez-Arcelus, R. P. Lach, C. Ingle, J. Nisbett, R. Guigo, and E. T. Dermitzakis. 2010. "Transcriptome genetics using second generation sequencing in a Caucasian population." *Nature* 464 (7289): 773–77.

Pickrell, J. K., J. C. Marioni, A. A. Pai, J. F. Degner, B. E. Engelhardt, E. Nkadori, J. B. Veyrieras, M. Stephens, Y. Gilad, and J. K. Pritchard. 2010. "Understanding mechanisms underlying human gene expression variation with RNA sequencing." *Nature* 464 (7289): 768–72.