Annex 3: EdgeR DE analysis

Juan Manuel Medina Méndez

```r
# Data reading
setwd("~/ProjectX")

# Library
library("edgeR")
library("limma")
library("ggplot2")
library("pheatmap")

# Data reading
matrix_plus <- read.table("EM.plus.tab", h = F, stringsAsFactors = F)
matrix_minus <- read.table("EM.minus.tab", h = F, stringsAsFactors = F)

# Removal of the SIZE column
matrix_plus <- matrix_plus[, -2]
matrix_minus <- matrix_minus[, -2]
matrix_control <- data.frame(matrix_plus[, c(1,2,3,4,5)], matrix_minus[, c(2,3,4,5)])
matrix_kd <- data.frame(matrix_plus[, c(1,6,7,8,9)], matrix_minus[, c(6,7,8,9)])

# Column renaming
names(matrix_plus) <- c("POSITION", "C0+", "C1+", "C2+", "C3+", "KD0+", "KD1+", "KD2+", "KD3+")
names(matrix_minus) <- c("POSITION", "C0-", "C1-", "C2-", "C3-", "KD0-", "KD1-", "KD2-", "KD3-")
names(matrix_control) <- c("POSITION", "C0+", "C1+", "C2+", "C3+", "C0-", "C1-", "C2-", "C3-")
names(matrix_kd) <- c("POSITION", "KD0+", "KD1+", "KD2+", "KD3+", "KD0-", "KD1-", "KD2-", "KD3-")

# Positions as rownames
cts_matrix_plus <- matrix_plus[, -1]
rownames(cts_matrix_plus) <- matrix_plus[, 1]
cts_matrix_minus <- matrix_minus[, -1]
rownames(cts_matrix_minus) <- matrix_minus[, 1]

cts_matrix_control <- matrix_control[, -1]
rownames(cts_matrix_control) <- matrix_control[, 1]
cts_matrix_kd <- matrix_kd[, -1]
rownames(cts_matrix_kd) <- matrix_kd[, 1]

# Metadata
intra_strand <- as.data.frame(c(rep("control", 4), rep("KD", 4)))
colnames(intra_strand) <- "class"
inter_strand <- as.data.frame(c(rep("plus", 4), rep("minus", 4)))
colnames(inter_strand) <- "class"

# Library sizes
matrices <- list(cts_matrix_plus, cts_matrix_minus, cts_matrix_control, cts_matrix_kd)
lapply(matrices, colSums)

# Distance matrix calculations over transposed and CPM-normalized EM
dist(t(scale(cts_matrix_plus, center = F, scale = colSums(cts_matrix_plus))))
dist(t(scale(cts_matrix_minus, center = F, scale = colSums(cts_matrix_minus))))
```

```
dist(t(scale(cts_matrix_control, center = F, scale = colSums(cts_matrix_control))))
dist(t(scale(cts_matrix_kd, center = F, scale = colSums(cts_matrix_kd))))

# Densities of expression values
plotDensities(log(cts_matrix_plus), legend = F)
```

```
plotDensities(log(cts_matrix_minus), legend = F)
```

```
plotDensities(log(cts_matrix_control), legend = F)
```

```
plotDensities(log(cts_matrix_kd), legend = F)
```

```
# Construction of DGEList objects
DGE_plus <- DGEList(counts = cts_matrix_plus)
DGE_minus <- DGEList(counts = cts_matrix_minus)
DGE_control <- DGEList(counts = cts_matrix_control)
DGE_kd <- DGEList(counts = cts_matrix_kd)

# Normalization factors to scale the library sizes by the mapped tags
DGE_plus <- calcNormFactors(DGE_plus, method = "TMM")
DGE_minus <- calcNormFactors(DGE_minus, method = "TMM")
DGE_control <- calcNormFactors(DGE_control, method = "TMM")
DGE_kd <- calcNormFactors(DGE_kd, method = "TMM")

# Calculate log CPM values: for PCAs and heatmaps
EM_plus <- cpm(DGE_plus, log = T)
EM_minus <- cpm(DGE_minus, log = T)
EM_control <- cpm(DGE_control, log = T)
EM_kd <- cpm(DGE_kd, log = T)

# Design matrices
design_control_kd <- model.matrix(~ class, data = intra_strand)
design_plus_minus <- model.matrix(~ class, data = inter_strand)

# PCA
pca_plus <- prcomp(x = t(EM_plus), scale = T, center = T)
pca_minus <- prcomp(x = t(EM_minus), scale = T, center = T)
pca_control <- prcomp(x = t(EM_control), scale = T, center = T)
pca_kd <- prcomp(x = t(EM_kd))

# Inspect components
summary(pca_plus)
summary(pca_minus)
summary(pca_control)
summary(pca_kd)

# Proportion of variance extraction
prop_var_plus <- summary(pca_plus)$importance[2, ]
prop_var_minus <- summary(pca_minus)$importance[2, ]
prop_var_control <- summary(pca_control)$importance[2, ]
prop_var_kd <- summary(pca_kd)$importance[2, ]

qplot(data = as.data.frame(pca_plus$x), x = PC1, y = PC2, color = intra_strand$class,
      label = rownames(intra_strand)) +
```

```
  labs(x = paste("PC1", round(prop_var_plus[1] * 100, digits = 2), "%", sep = " "),
       y = paste("PC2", round(prop_var_plus[2] * 100, digits = 2), "%", sep = " ")) +
  ggtitle("PCA (control & KD: plus strand)")
```

```
qplot(data = as.data.frame(pca_minus$x), x = PC1, y = PC2, color = intra_strand$class,
      label = rownames(intra_strand)) +
  labs(x = paste("PC1", round(prop_var_minus[1] * 100, digits = 2), "%", sep = " "),
       y = paste("PC2", round(prop_var_minus[2] * 100, digits = 2), "%", sep = " ")) +
  ggtitle("PCA (control & KD: minus strand)")
```

```
qplot(data = as.data.frame(pca_control$x), x = PC1, y = PC2, color = inter_strand$class,
      label = rownames(inter_strand)) +
  labs(x = paste("PC1", round(prop_var_control[1] * 100, digits = 2), "%", sep = " "),
       y = paste("PC2", round(prop_var_control[2] * 100, digits = 2), "%", sep = " ")) +
  ggtitle("PCA (controls of both strands)")
```

```
qplot(data = as.data.frame(pca_kd$x), x = PC1, y = PC2, color = inter_strand$class,
      label = rownames(inter_strand)) +
  labs(x = paste("PC1", round(prop_var_kd[1] * 100, digits = 2), "%", sep = " "),
       y = paste("PC2", round(prop_var_kd[2] * 100, digits = 2), "%", sep = " ")) +
  ggtitle("PCA (KDs of both strands)")
```

```
# Heatmaps
pheatmap(mat = EM_plus, kmeans_k = 50, scale = "row")
```

```
pheatmap(mat = EM_minus, kmeans_k = 50, scale = "row")
```

```
pheatmap(mat = EM_control, kmeans_k = 50, scale = "row")
```

```
# Maximization of the negative binomial LH to calculate the common, trended and tagwise dispersion
# across all tags
disp_plus <- estimateDisp(DGE_plus, design_control_kd)
disp_minus <- estimateDisp(DGE_minus, design_control_kd)
disp_control <- estimateDisp(DGE_control, design_plus_minus)
disp_kd <- estimateDisp(DGE_kd, design_plus_minus)

# Plotting the dispersion
plotBCV(y = disp_plus)
```

```
plotBCV(y = disp_minus)
```

```
plotBCV(y = disp_control)
```

```
plotBCV(y = disp_kd)
```

```
# Fit quasi-LH negative binomial GLM to each DHSs (count data)
fit_plus <- glmQLFit(y = disp_plus, design = design_plus_minus)
fit_minus <- glmQLFit(y = disp_minus, design = design_plus_minus)
fit_control <- glmQLFit(y = disp_control, design = design_control_kd)
fit_kd <- glmQLFit(y = disp_kd, design = design_control_kd)

# QL F-test
qlf_plus <- glmQLFTest(fit_plus, coef = 2)
qlf_minus <- glmQLFTest(fit_minus, coef = 2)
qlf_control <- glmQLFTest(fit_control, coef = 2)
qlf_kd <- glmQLFTest(fit_kd, coef = 2)
```

```r
# Extract results
res_qlf_plus <- as.data.frame(qlf_plus$table)
res_qlf_minus <- as.data.frame(qlf_minus$table)
res_qlf_control <- as.data.frame(qlf_control$table)
res_qlf_kd <- as.data.frame(qlf_kd$table)

# Summarize number of up and down regulated DHSs at 5% FDR
is_de_plus <- decideTestsDGE(qlf_plus, p.value = 0.05)
summary(is_de_plus)
is_de_minus <- decideTestsDGE(qlf_minus, p.value = 0.05)
summary(is_de_minus)
is_de_control <- decideTestsDGE(qlf_control, p.value = 0.05)
summary(is_de_control)
is_de_kd <- decideTestsDGE(qlf_kd, p.value = 0.05)
summary(is_de_kd)

# Plot LFC against log-counts per million, with DE DHSs highlighted
# Green lines indicating 2 LFC
plotMD(qlf_plus, main = "MA plot (control & KD: plus strand)")
abline(h = c(-1, 1), col = "green")
```

```r
plotMD(qlf_minus, main = "MA plot (control & KD: minus strand)")
abline(h = c(-1, 1), col = "green")
```

```r
plotMD(qlf_control, main = "MA plot (controls of both strands)")
abline(h = c(-1, 1), col = "green")
```

```r
plotMD(qlf_kd, main = "MA plot (KDs of both strands)")
abline(h = c(-1, 1), col = "green")
```

```r
# Customized volcano-plots
plot(-log10(PValue) ~ logFC, data = topTags(qlf_plus, n = Inf, sort.by = "none"),
     col = factor(decideTestsDGE(qlf_plus) != 0), pch = 16,
     main = "Volcano plot \n (control & KD: plus strand)")
```

```r
plot(-log10(PValue) ~ logFC, data = topTags(qlf_minus, n = Inf, sort.by = "none"),
     col = factor(decideTestsDGE(qlf_minus) != 0), pch = 16,
     main = "Volcano plot \n (control & KD: minus strand)")
```

```r
plot(-log10(PValue) ~ logFC, data = topTags(qlf_control, n = Inf, sort.by = "none"),
     col = factor(decideTestsDGE(qlf_control) != 0), pch = 16,
     main = "Volcano plot \n (controls of both strands)")
```

```r
plot(-log10(PValue) ~ logFC, data = topTags(qlf_kd, n = Inf, sort.by = "none"),
     col = factor(decideTestsDGE(qlf_kd) != 0), pch = 16,
     main = "Volcano plot \n (KDs of both strands)")
```