— Differential Gene Expression —

Anja von Heydebreck, Manuela Hummel

March 2, 2008

1. **Loading Data.** The data used for these exercises come from a study of Chiaretti et al. (Blood 103:2771-8, 2004) on acute lymphoblastic leukemia (ALL), which was conducted with HG-U95Av2 Affymetrix arrays. The data package ALL contains an ExpressionSet object called ALL, which contains the expression data that were normalized with rma (intensities are on the log2-scale), and annotations of the samples.

   (a) Load the ALL package. What is the dimension of the expression data matrix?

   ```
   > library(ALL)
   > library(hgu95av2)
   > library(annotate)
   > data(ALL)
   > dim(exprs(ALL))
   ```

   (b) Use the function show to get an overview of the ExpressionSet object. What are the variables describing the samples stored in the pData slot?

   ```
   > show(ALL)
   > print(summary(pData(ALL)))
   ```

2. **B-cell ALL.** We want to look at the B-cell ALL samples (they can be identified by the column BT of the pData slot of the ExpressionSet ALL). Of particular interest is the comparison of samples with the BCR/ABL fusion gene resulting from a translocation of the chromosomes 9 and 22 (labelled BCR/ABL in the column mol), with samples that are cytogenetically normal (labelled NEG).

   Define an ExpressionSet object containing only the data from the B-cell ALL samples. How many samples belong to the cytogenetically defined groups?

   ```
   > pdat <- pData(ALL)
   > table(pdat$BT)
   > table(pdat$mol)
   > subset <- intersect(grep("^B", as.character(pdat$BT)),
   +     which(as.character(pdat$mol) %in% c("BCR/ABL", "NEG")))
   > eset <- ALL[, subset]
   > table(eset$mol)
   ```

3. **Non-specific filtering.** Many of the genes on the chip won't be expressed in the B-cell lymphozytes studied here, or might have only small variability across the samples.

   We try to remove these genes (more precisely: the corresponding probe sets) with an intensity filter (the intensity of a gene should be above 100 in at least 25 percent of the samples), and a variance filter (the interquartile range of log2-intensities should be at least 0.5). We create a new ExpressionSet containing only the probe sets which passed our filter. How many probe sets do we get?

   ```
   > library(genefilter)
   > f1 <- pOverA(0.25, log2(100))
   > f2 <- function(x) (IQR(x) > 0.5)
   ```

```
> ff <- filterfun(f1, f2)
> selected <- genefilter(eset, ff)
> sum(selected)
> esetSub <- eset[selected, ]
```

4. **Differential expression.** Now we are ready to examine the selected genes for differential expression between the BCR/ABL samples and the cytogenetically normal ones.

   (a) Use the two-sample t-test to identify genes that are differentially expressed between the two groups. The function `mt.teststat` from the `multtest` package allows to compute several commonly used test statistics for all rows of a data matrix – study its help page. First, we calculate the nominal p-values – the function `pt` gives the distribution function of the t-distribution. We can get an impression of the amount of differential gene expression by looking at a histogram of the p-value distribution.

   ```
   > library(multtest)
   > cl <- as.numeric(esetSub$mol == "BCR/ABL")
   > t <- mt.teststat(exprs(esetSub), classlabel = cl, test = "t.equalvar")
   > pt <- 2 * pt(-abs(t), df = ncol(exprs(esetSub)) - 2)
   > hist(pt, 50)
   ```

   (b) The second possibility is to conduct a permutation test for each gene. The function `mt.maxT` computes permutation p-values (`rawp`) and FWER-adjusted p-values by the Westfall-Young method (`adjp`) at the same time. How many FWER-adjusted p-values are smaller than 0.1?

   ```
   > mT <- mt.maxT(exprs(esetSub), classlabel = cl, B = 1000)
   > sum(mT$adjp < 0.1)
   ```

   A higher number of permutations (given by `B`) would be preferable, but would take more time. Note: The function returns the p-values ordered by their size. To get them in the original order, do the following:

   ```
   > pPermRaw <- mT$rawp[order(mT$index)]
   > pWestfallYoung <- mT$adjp[order(mT$index)]
   ```

   (c) Plot the p-values against the log-ratios (differences of mean log-intensities within the two groups) in a *volcano plot*. Note the asymmetry of the volcano plot. Compare the permutation p-values to those from the parametric t-test. Why do we obtain more extremely small p-values with the parametric test?

   ```
   > hist(pt, 50)
   > logRatio <- rowMeans(exprs(esetSub)[, cl == 1]) - rowMeans(exprs(esetSub)[,
   +     cl == 0])
   > plot(logRatio, -log10(pt), xlab = "log-ratio", ylab = "-log10(p)")
   > plot(log10(pt), log10(pPermRaw), main = "log10(p-value)",
   +     xlab = "parametric", ylab = "permutation test")
   ```

   (d) The function `mt.rawp2adjp` from the `multtest` package contains different multiple testing procedures. Look at the help page of this function. For p-value adjustment in terms of the FDR, we use the method of Benjamini and Hochberg. How many genes do you get when imposing an FDR of 0.1?

   ```
   > pAdjusted <- mt.rawp2adjp(pt, proc = c("BH"))
   > sum(pAdjusted$adjp[, "BH"] < 0.1)
   ```

   Also this function returns the adjusted p-values ordered from the smallest to the largest. To obtain the original ordering, we do:

   ```
   > pBH <- pAdjusted$adjp[order(pAdjusted$index), "BH"]
   ```

5. **Annotation.**

(a) Now we want to see which genes are the most significant ones, and look at their raw and adjusted p-values from the different methods. Gene symbols are provided in the annotation package hgu95av2.

```
> diff <- pAdjusted$index[1:10]
> genesymbols <- mget(featureNames(esetSub)[diff], hgu95av2SYMBOL)
> pvalues <- cbind(pt, pPermRaw, pWestfallYoung, pBH)[diff,
+     ]
> colnames(pvalues) <- c("pt", "pPermRaw", "pWestfallYoung",
+     "pBH")
> rownames(pvalues) <- genesymbols
> print(pvalues)
```

(b) The top 3 probe sets represent the ABL1 gene, which is affected by the translocation characterizing the BCR/ABL samples. Now we want to see whether there are further probe sets representing this gene, and whether these have been selected by our non-specific filtering.

```
> geneSymbols = mget(featureNames(ALL), hgu95av2SYMBOL)
> ABL1probes <- which(geneSymbols == "ABL1")
> selected[ABL1probes]
```

(c) So the other probe sets representing ABL1 have been filtered out because of low intensities or low variance. Now we want to see whether they also indicate differential expression of the ABL1 gene.

```
> tABL1 <- mt.teststat(exprs(eset)[ABL1probes, ], classlabel = cl,
+     test = "t.equalvar")
> ptABL1 <- 2 * pt(-abs(tABL1), df = ncol(exprs(esetSub)) -
+     2)
> sort(ptABL1)
```

We see that only three out of the six ABL1 probe sets show evidence (in fact, very strong evidence) for differential expression! It might be interesting to further investigate the ABL1 probe sets regarding e.g. their location in the ABL1 transcript sequence – indeed the BCR/ABL fusion gene resulting from the translocation differs from the normal ABL1 gene.

6. **Gene Ontology.**

(a) Many of the effects due to the BCR/ABL translocation are mediated by tyrosine kinase activity. Let's look at the probe sets that are annotated at the GO term `protein-tyrosine kinase activity`, which has the identifier GO:0004713.

```
> gN <- featureNames(esetSub)
> tykin <- unique(unlist(lookUp("GO:0004713", "hgu95av2",
+     "GO2ALLPROBES")))
> str(tykin)
> sel <- (gN %in% tykin)
```

(b) We can now check whether there are more differentially expressed genes among the tyrosine kinases than among the other genes. Fisher's exact test for contingency tables is used to check whether the proportions of differentially expressed genes are significantly different in the two gene groups.

```
> tab <- table(pt < 0.05, sel, dnn = c("p < 0.05", "tykin"))
> print(tab)
> fisher.test(tab)
```

7. **Limma.** A t-test analysis can also be conducted with functions of the `limma` package.

(a) First, we have to define the design matrix. One possibility is to use an intercept term that represents the mean log-intensity of a gene across all samples (first column consisting of 1's), and to encode the difference between the two classes in the seond column.

```
> library(limma)
> design <- cbind(mean = 1, diff = cl)
```

(b) A linear model is fitted for every gene by the function `lmFit`, and Empirical Bayes moderation of the standard errors is done by the function `eBayes`.

```
> fit <- lmFit(exprs(esetSub), design)
> fit2 <- eBayes(fit)
> topTable(fit2, coef = "diff", adjust.method = "fdr")
```

(c) When you compare the resulting p-values with those from the parametric t-test (Exercise 4.a), you will see that they are almost identical. Because of the large number of samples, the Empirical Bayes moderation is not so relevant in this data set – the gene-specific variance can well be estimated from the data of each gene.

```
> plot(log10(pt), log10(fit2$p.value[, "diff"]), xlab = "two-sample t-test",
+     ylab = "limma")
> abline(c(0, 1), col = "Red")
```

8. **ROC curve screening.**

(a) We want to find marker genes that are specifically expressed in leukemias with the BCR/ABL-translocation. At a specificity of at least 0.9, we would like to identify the genes with the best sensitivity for the BCR/ABL phenotype. This can be expressed by the partial area under the ROC curve (`pAUC`, we choose $t0 = 0.1$). To limit the computation time, we compute the `pAUC`-statistic only for the first 100 probe sets.

```
> library(ROC)
> mypauc1 <- function(x) {
+     pAUC(rocdemo.sca(truth = cl, data = x, rule = dxrule.sca),
+         t0 = 0.1)
+ }
> pAUC1s <- esApply(esetSub[1:100, ], 1, mypauc1)
```

(b) Select the two probe sets with the maximal value of our `pAUC`-statistic, and plot the corresponding ROC curves. Look for a comparison at the t-test p-values for these genes.

```
> best <- order(pAUC1s, decreasing = T)[1:2]
> x11()
> par(mfrow = c(1, 2))
> for (pS in best) {
+     RC <- rocdemo.sca(truth = cl, data = exprs(esetSub)[pS,
+         ], rule = dxrule.sca)
+     plot(RC, main = featureNames(esetSub)[pS])
+ }
> print(pt[best])
```