

---

## Laboratory 2:2 Topics in analysis of microarray data: differential expression, test statistics and multiple testing

Day 2: Day August 17, 2004: 13:45 – 15:15

Ben Bolstad, Biostatistics, University of California, Berkeley

---

### Key Concepts

- Finding differentially expressed genes
- Adjusting p-values for multiple testing

### What you will be able to do at end of this section

- use BioC software to carry out multiple hypothesis testing adjustment
- Identify differential genes using moderated test-statistics.

### Introduction

This lab will give you the opportunity to use moderated test statistics to detect differential expression. Using simulations you will observe what happens when you carry out multiple tests and the effect of different methods of adjusting P-values.

### Multiple testing

First load the multiple testing library. Type

```
library(multtest)
```

Next we write a function to extract the p-values from a simple t-test. Type

```
t.stat.pvalue <- function(X) {  
  t.test(X)$p.value  
}
```

Next we simulate log<sub>2</sub> fold-changes for 10000 genes on 6 arrays. Note in this simulation there are no genes that have any true differential expression.

```
X <- matrix(rnorm(60000), nrow=10000)
```

Now apply a t-test to each row. Type

```
X.pvalues <- apply(X,1,t.stat.pvalue)
```

Now count how many hypothesis tests rejected the null hypothesis (ie how many false positives).

Type

```
sum(X.pvalues < 0.05)
```

Now carry out a Bonferroni adjustment on the raw P-values. Do this by typing

```
X.pvalues.bonferroni <- mt.rawp2adjp(X.pvalues,proc="Bonferroni")
X.pvalues.bonferroni <-
X.pvalues.bonferroni$adjp[order(X.pvalues.bonferroni$index),2]
```

How many false positives do you expect now? Check by typing

```
sum(X.pvalues.bonferroni < 0.05)
```

Repeat the same process using the Benjamini-Hochberg FDR controlling adjustment.

```
X.pvalues.bh <- mt.rawp2adjp(X.pvalues,proc="BH")
X.pvalues.bh <- X.pvalues.bh$adjp[order(X.pvalues.bh$index),2]
sum(X.pvalues.bh < 0.05)
```

Now lets carry out a simulation where there are differentially expressed genes. First simulate 9000 non differential genes on 5 arrays and 1000 differential genes on 5 arrays. Do this by typing

```
num.true <- 9000
num.false <- 1000
num <- num.true + num.false
non.diff.mean <- 0
diff.mean <- 2
means <- c(rep(non.diff.mean,num.true),rep(diff.mean,num.false))
nsamples <- 5
Condition2 <-
matrix(rnorm(nsamples*num,rep(means,nsamples)),num,nsamples)
```

Next lets carry out a t-test for each gene and store the p-values. Do this by typing

```
Condition2.t <-
apply(Condition2,1,mean)/sqrt(apply(Condition2,1,var)/nsamples)
Condition2.pval <- (1-pt(Condition2.t,nsamples))
```

Suppose we went to test at the 5% level of significance. Using the raw p-values how many false positives and how many false negatives do you observe? Find out by typing

```
FalsePositives <- sum(Condition2.pval[1:num.true] < 0.05)
FalseNegatives <- sum(Condition2.pval[(num.true+1):num] > 0.05)
FalsePositives;FalseNegatives
```

Next use the Bonferroni procedure to control the FWER. How many false positives and false negatives do you observe now? Find out by typing

```
FalsePositivesBonferroni <- sum(Condition2.pval[1:num.true] < 0.05/num)
FalseNegativesBonferroni <- sum(Condition2.pval[(num.true+1):num] >
0.05/num)
FalsePositivesBonferroni;FalseNegativesBonferroni
```

How does this compare to the number of false positives and false negatives you observed using the raw p-values?

Next lets use the Benjamini and Hochberg procedure to control the FDR.

```
FDR <- 0.05
cutoff <- max(sort(Condition2.pval)[sort(Condition2.pval) <=
(1:num)/num*FDR])
FalsePositivesFDR <- sum(Condition2.pval[1:num.true] < cutoff)
FalseNegativesFDR <- sum(Condition2.pval[(num.true+1):num] > cutoff)
FalsePositivesFDR;FalseNegativesFDR
```

How do the number of false positives and false negatives you observe compare to the two earlier results.

## Testing for differential expression

First we will load the *limma* library and the data. Note that you will need to load the data from a website. Do this by typing

```
library(limma)
sourceURL("http://www.stat.berkeley.edu/~bolstad/ApoAI.RData")
```

In this section we consider a case study where two RNA sources are compared through a common reference RNA. The analysis of the log-ratios involves a two-sample comparison of means for each gene.

**Background.** The data is from a study of lipid metabolism by Callow et al (2000) ([1]). The apolipoprotein AI (ApoAI) gene is known to play a pivotal role in high density lipoprotein (HDL) metabolism. Mice which have the ApoAI gene knocked out have very low HDL cholesterol levels. The purpose of this experiment is to determine how ApoAI deficiency affects the action of other genes in the liver, with the idea that this will help determine the molecular pathways through which ApoAI operates.

**Hybridizations.** The experiment compared 8 ApoAI knockout mice with 8 wild type (normal) C57BL/6 ("black six") mice, the control mice. For each of these 16 mice, target mRNA was obtained from liver tissue and labelled using a Cy5 dye. The RNA from each mouse was hybridized to a separate microarray. Common reference RNA was labelled with Cy3 dye and used for all the arrays. The reference RNA was obtained by pooling RNA extracted from the 8 control mice.

<i>Number of arrays</i>	<i>Red (Cy5)</i>	<i>Green (Cy3)</i>
8	Wild Type "black six" mice (WT)	Pooled Reference (Ref)
8	ApoAI Knockout (KO)	Pooled Reference (Ref)

As before we will pre-process the data. Note that this is a slightly different function than we used in the preprocessing lab, but it does pretty much the same thing.

```
MA <- normalizeWithinArrays(RG)
```

Next we use a linear model to estimate the difference in expression between the WT and KO mice. First we set up design matrix. Then we fit the linear model to each gene.

```
design <- matrix(c(rep(1,16),rep(0,8),rep(1,8)),ncol=2)
colnames(design) <- c("WT-Ref","KO-WT")
design
fit <- lmFit(MA,design=design)
names(fit)
```

Next we write a function to create a table of the top ranked genes when ranked by the unmoderated t-statistic. Do this by typing

```

my.toptable <- function(x,n=10){
  unmoderated.t <- x$coefficients[,2]/(x$sigma*0.5)
  unmoderated.p <- 2*(1-pt(abs(unmoderated.t),x$df.residual))

  ordering <- order(-abs(unmoderated.t[rank(-abs(unmoderated.t)) <=
n]))
  cbind(x$genes,M=x$coefficients[,2],A=x$Amean,unmoderated.t,unmodera
ted.p)[rank(-abs(unmoderated.t)) <= n,][ordering,]
}

```

Next list the top 20 genes

```
my.toptable(fit,n=20)
```

Now we will compute the moderated test statistic. As mentioned in the lecture this is computed by using an empirical bayes procedure.

```
fit2 <- eBayes(fit)
names(fit2)
```

Finally, look at a table of the top 20 genes

```
topTable(fit2,coef="KO-WT",n=20,adjust="none")
```

Do you notice any difference between this table and the previous table? Why do you think this is the case?

## Appendix

### 1. Resources

#### i) Original Papers

- Smyth, G. K. (2004). Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology* 3, No. 1, Article 3.
- Y. Ge, S. Dudoit & T. P. Speed (2003), Resampling-based multiple testing for microarray data hypothesis *Test* 12(1) : 1-44
- M. J. Callow, S. Dudoit, E. L. Gong, T. P. Speed, and E. M. Rubin (2000). Microarray expression profiling identifies genes with altered expression in HDL deficient mice. *Genome Research*, Vol. 10, No. 12, p. 2022-2029.

- S. Dudoit, Y. H. Yang, T. P. Speed, and M. J. Callow (2002). Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statistica Sinica*, Vol. 12, No. 1, p. 111-139.

**ii) Software**

- [www.bioconductor.org](http://www.bioconductor.org)

**iii) Web Sites:**

- <http://bioinf.wehi.edu.au/limma/>

-

**2. Demonstration Overheads**

© 2004 Canadian Genetic Diseases Network