EFDR: Some statistical methods for the identification of differentially expressed genes

June 6, 2003

Introduction

In this lab we introduce you to some extra statistical methods for the identification of differentially expressed genes between two populations. This lab is a complement of Lab 4, but relies upon some ideas that come from wavelet analysis.

```
> library(Biobase)

Welcome to Bioconductor
         To view some introductory material -- look at our vignettes
         Simply type: openVignette()
         to see the available vignettes
         To read a vignette see the openVignette help page for details
Creating a new generic function for "summary" in package
Biobase

> library(golubEsets)
> library(ctest)
> library(multtest)
> library(bioclabs)
> library(modreg)
> library(wavethresh)

S/R wavelet software, release 2.2, installed
Copyright Guy Nason 1993
R version 2.2-7: Arne Kovac 1997; Martin Maechler, 1999-2001

> library(Milan)
```

```
Loading required package: dr

Attaching package 'Milan':


        The following object(s) are masked from package:dr :

         dr dr.z
```

The lab comes in two parts, in the first we consider how wavelet and thresholding procedures may be used for finding genes that are differentially expressed, while in the second part we consider FDR based procedures that are available in R.

We use expression data from an experiment where sixteen genes were spiked in at different known concentrations in different hybridizations and are thus differentially expressed. Expression measures are stored in an `exprSet` object available through the *bioclabs* package.

```
> data("eset12")
> genenames <- colnames(pData(eset12))[-1]
```

The concentrations of the sixteen genes in each of 24 hybridizations are stored in the `phenoData` slot of the `exprSet` object `eset12`.

```
> pData(eset12)
```

	population	37777_at	684_at	1597_at	38734_at	39058_at	36311_at
1521m99hpp_av06	0	512	1024	0.00	0.25	0.5	1
1521n99hpp_av06	0	512	1024	0.00	0.25	0.5	1
1521o99hpp_av06	0	512	1024	0.00	0.25	0.5	1
1521p99hpp_av06	0	512	1024	0.00	0.25	0.5	1
1521q99hpp_av06	1	1024	0	0.25	0.50	1.0	2
1521r99hpp_av06	1	1024	0	0.25	0.50	1.0	2
1521s99hpp_av06	1	1024	0	0.25	0.50	1.0	2
1521t99hpp_av06	1	1024	0	0.25	0.50	1.0	2
1532m99hpp_av04	0	512	1024	0.00	0.25	0.5	1
1532n99hpp_av04	0	512	1024	0.00	0.25	0.5	1
1532o99hpp_av04	0	512	1024	0.00	0.25	0.5	1
1532p99hpp_av04	0	512	1024	0.00	0.25	0.5	1
1532q99hpp_av04	1	1024	0	0.25	0.50	1.0	2
1532r99hpp_av04	1	1024	0	0.25	0.50	1.0	2
1532s99hpp_av04	1	1024	0	0.25	0.50	1.0	2
1532t99hpp_av04r	1	1024	0	0.25	0.50	1.0	2
2353m99hpp_av08	0	512	1024	0.00	0.25	0.5	1
2353n99hpp_av08	0	512	1024	0.00	0.25	0.5	1

2353o99hpp_av08	0	512	1024	0.00	0.25	0.5	1
2353p99hpp_av08	0	512	1024	0.00	0.25	0.5	1
2353q99hpp_av08	1	1024	0	0.25	0.50	1.0	2
2353r99hpp_av08	1	1024	0	0.25	0.50	1.0	2
2353s99hpp_av08	1	1024	0	0.25	0.50	1.0	2
2353t99hpp_av08	1	1024	0	0.25	0.50	1.0	2

	36889_at	1024_at	36202_at	36085_at	40322_at	407_at	1091_at
1521m99hpp_av06	2	4	8	16	32	512	128
1521n99hpp_av06	2	4	8	16	32	512	128
1521o99hpp_av06	2	4	8	16	32	512	128
1521p99hpp_av06	2	4	8	16	32	512	128
1521q99hpp_av06	4	8	16	32	64	1024	256
1521r99hpp_av06	4	8	16	32	64	1024	256
1521s99hpp_av06	4	8	16	32	64	1024	256
1521t99hpp_av06	4	8	16	32	64	1024	256
1532m99hpp_av04	2	4	8	16	32	512	128
1532n99hpp_av04	2	4	8	16	32	512	128
1532o99hpp_av04	2	4	8	16	32	512	128
1532p99hpp_av04	2	4	8	16	32	512	128
1532q99hpp_av04	4	8	16	32	64	1024	256
1532r99hpp_av04	4	8	16	32	64	1024	256
1532s99hpp_av04	4	8	16	32	64	1024	256
1532t99hpp_av04r	4	8	16	32	64	1024	256
2353m99hpp_av08	2	4	8	16	32	512	128
2353n99hpp_av08	2	4	8	16	32	512	128
2353o99hpp_av08	2	4	8	16	32	512	128
2353p99hpp_av08	2	4	8	16	32	512	128
2353q99hpp_av08	4	8	16	32	64	1024	256
2353r99hpp_av08	4	8	16	32	64	1024	256
2353s99hpp_av08	4	8	16	32	64	1024	256
2353t99hpp_av08	4	8	16	32	64	1024	256

	1708_at	33818_at	546_at
1521m99hpp_av06	256	32	8
1521n99hpp_av06	256	32	8
1521o99hpp_av06	256	32	8
1521p99hpp_av06	256	64	8
1521q99hpp_av06	512	64	16
1521r99hpp_av06	512	64	16
1521s99hpp_av06	512	64	16
1521t99hpp_av06	512	128	16
1532m99hpp_av04	256	32	8
1532n99hpp_av04	256	32	8

```
1532o99hpp_av04      256       32       8
1532p99hpp_av04      256       64       8
1532q99hpp_av04      512       64      16
1532r99hpp_av04      512       64      16
1532s99hpp_av04      512       64      16
1532t99hpp_av04r     512      128      16
2353m99hpp_av08      256       32       8
2353n99hpp_av08      256       32       8
2353o99hpp_av08      256       32       8
2353p99hpp_av08      256       64       8
2353q99hpp_av08      512       64      16
2353r99hpp_av08      512       64      16
2353s99hpp_av08      512       64      16
2353t99hpp_av08      512      128      16
```

Notice there are two populations and 12 replicates in each. We are interested in identifying genes that are differentially expressed in these two populations.

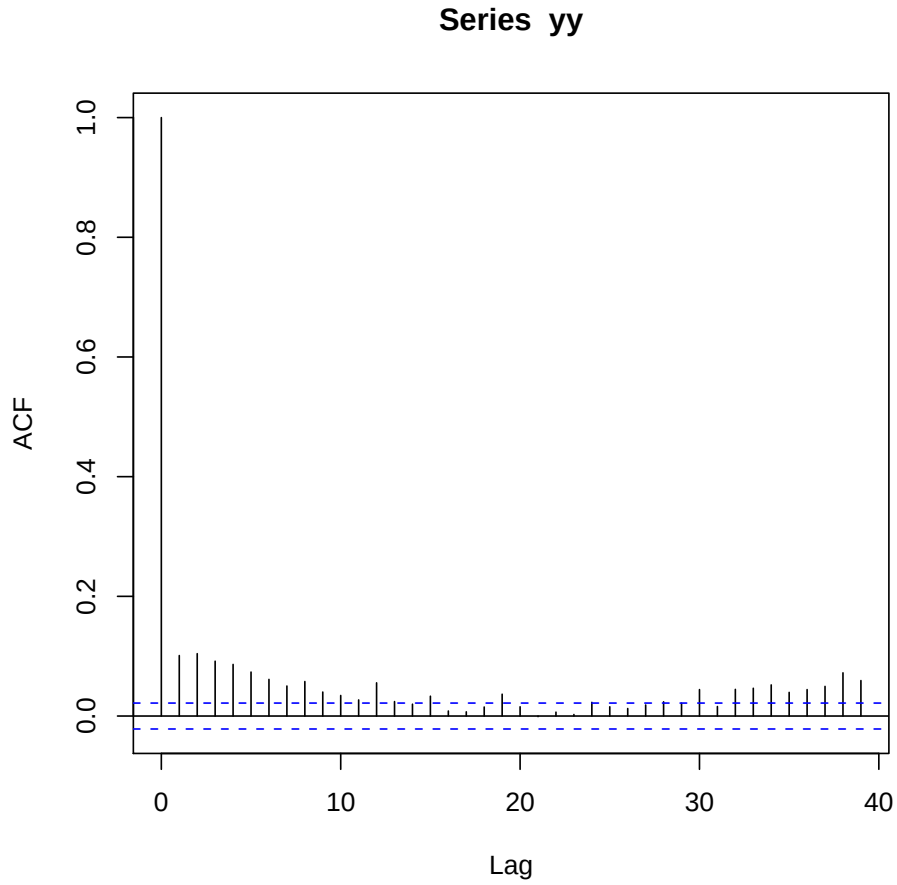
Let us create a matrix containing for each of the 12626 genes on the HGU95a chip (note this really is A and not Av2), its *A-value* or average log intensity, its *M-value* or difference of log intensities (log ratio), its two-sample t-statistic, and its nominal p-value from the t-distribution. The rows of the `scores` matrix correspond to genes and the columns to the four different types of statistics.

The data have already been transformed to the log scale. Base 2 logarithms were used.

```
> Index1 <- which(eset12$population == 0)
> Index2 <- which(eset12$population == 1)
> scores <- esApply(eset12, 1, function(x) {
+     tmp <- t.test(x[Index2], x[Index1], var.equal = TRUE)
+     c(mean(tmp$estimate), -diff(tmp$estimate), tmp$statistic,
+         tmp$p.value)
+ })
> scores <- t(scores)
> colnames(scores) <- c("A", "M", "t.stat", "p.value")
```

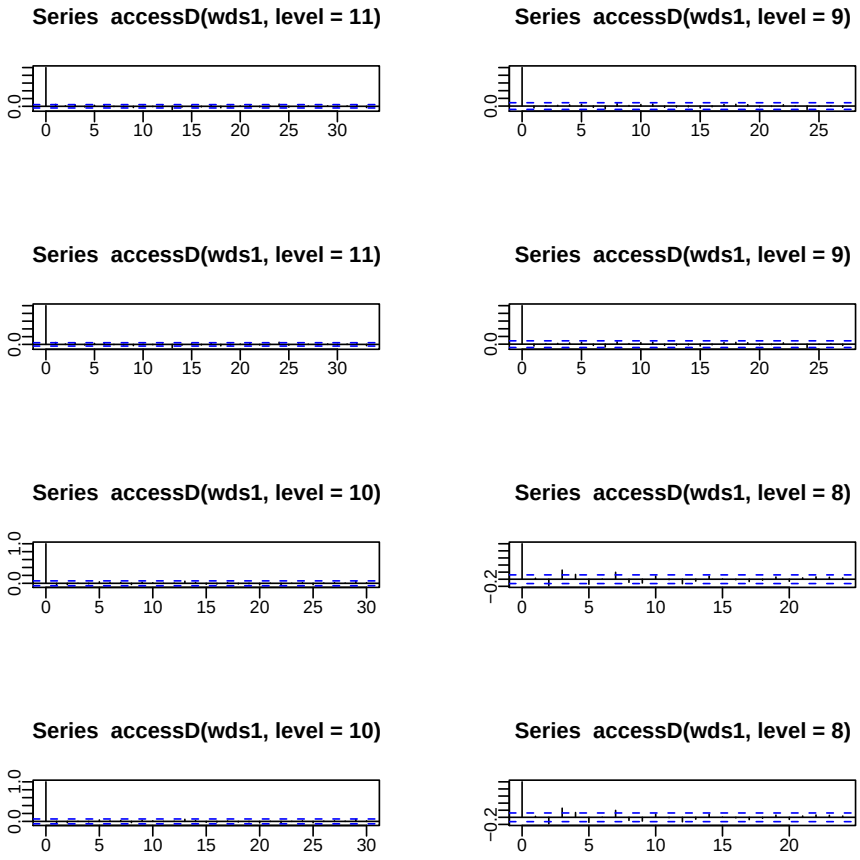
Before further discussing procedures designed to find differentially expressed genes, let us illustrate first the decorrelation properties of the wavelet transform, which may enhance multiple hypothesis based procedures if they are applied in the wavelet domain. We select first a subseries of length 8192, a power of 2, from the 12626 M-values. We next display an autocorrelation plot of the original M-values series, followed by autocorrelation plots of their wavelet coefficients at several scales. Notice the decorrelation properties of the discrete wavelet transform.

```
> yy <- scores[1:8192, 2]
> plot(acf(yy))
```



Let us look now the acf of the wavelet coefficients of the DWT of the previous series at several scales.

```
> wds1 <- wd(yy)
> par(mfcol = c(4, 2), mgp = c(5, 0.4, 0))
> plot(acf(accessD(wds1, level = 11)))
> plot(acf(accessD(wds1, level = 10)))
> plot(acf(accessD(wds1, level = 9)))
> plot(acf(accessD(wds1, level = 8)))
> plot(acf(accessD(wds1, level = 7)))
> plot(acf(accessD(wds1, level = 6)))
> plot(acf(accessD(wds1, level = 5)))
> plot(acf(accessD(wds1, level = 4)))
```



The above suggests that wavelet decompostion may enhance FDR multiple hypothesis testing procedures by coming closer to independance and by reducing efficiently the number of hypotheses being tested.

We are now going to use the empirical Bayes thresholding procedure to find differentially expressed genes. There are 12 replicates and we can use the t-test statistics to compare the gene expressions taking into account their variability. With 12 replicates but so many genes we expect a large number of false positives and we already saw in a previous Lab that t-tests were not successful in identifying the sixteen known genes. The idea is to use the t-scores as noisy observations and a natural approach to this problem, taking into account the sparsity of significant scores is thresholding: if the absolute value of a particular score Y_i exceeds some threshold t then it is taken to correspond to a nonzero signal which is then estimated, most simply by Y_i itself. If $|Y_i| < t$ then the corresponding gene expression is estimated to be zero.

```
> y <- scores[, 3]
> fit <- ebayesthresh(y, prior = "cauchy", a = NA, bayesfac = T,
+     sdev = NA, verbose = F, threshrule = "median")
```

```
> aux <- abs(fit)
> auxsv <- aux > 0
> cat("Number of signicant genes: \n")

Number of signicant genes:

> sum(as.integer(auxsv))

[1] 39

> rownames(scores[aux > 0, ])

 [1] "1024_at"    "1032_at"    "1091_at"    "1552_i_at"  "1708_at"
 [6] "216_at"     "286_at"     "31617_at"   "32115_r_at" "32650_at"
[11] "32660_at"   "32955_at"   "33264_at"   "33818_at"   "34246_at"
[16] "35339_at"   "35939_s_at" "36085_at"   "36176_at"   "36202_at"
[21] "36311_at"   "36664_at"   "36889_at"   "37307_at"   "37492_at"
[26] "37777_at"   "38254_at"   "38406_f_at" "38502_at"   "38734_at"
[31] "39058_at"   "39386_at"   "40322_at"   "40460_s_at" "407_at"
[36] "41184_s_at" "41764_at"   "546_at"     "966_at"
```

Let's see how we fared with the sixteen known to be differentially expressed genes.

```
> genenames

 [1] "37777_at" "684_at"   "1597_at"  "38734_at" "39058_at" "36311_at"
 [7] "36889_at" "1024_at"  "36202_at" "36085_at" "40322_at" "407_at"
[13] "1091_at"  "1708_at"  "33818_at" "546_at"
```

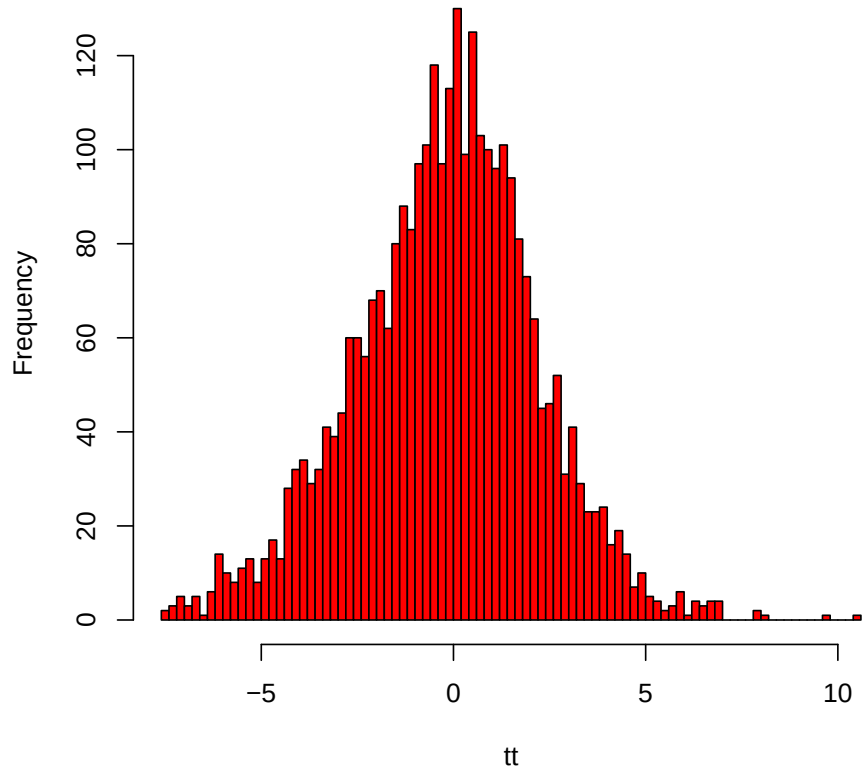
The examples in this second part will rely on the data reported in **?**. The basic comparisons here are between B-cell derived ALL, T-cell derived ALL and AML.

What follows serves as a short tutorial on the q-value, as well as instructions for how to use the q-value functions developed by Storey (2003) and available for use in R. The q-value is similar to the well known p-value. It gives each hypothesis test a measure of significance in terms of a certain error rate. The p-value of a test measures the minimum false positive rate that is incurred when calling that test significant. Likewise, the q-value of a test measures the minimum false discovery rate that is incurred when calling that test significant.

We first load the golub data set.

```
> data(golub)
> classlabel <- golub.cl
> dat = golub
> o <- rep(T, nrow(dat))
```

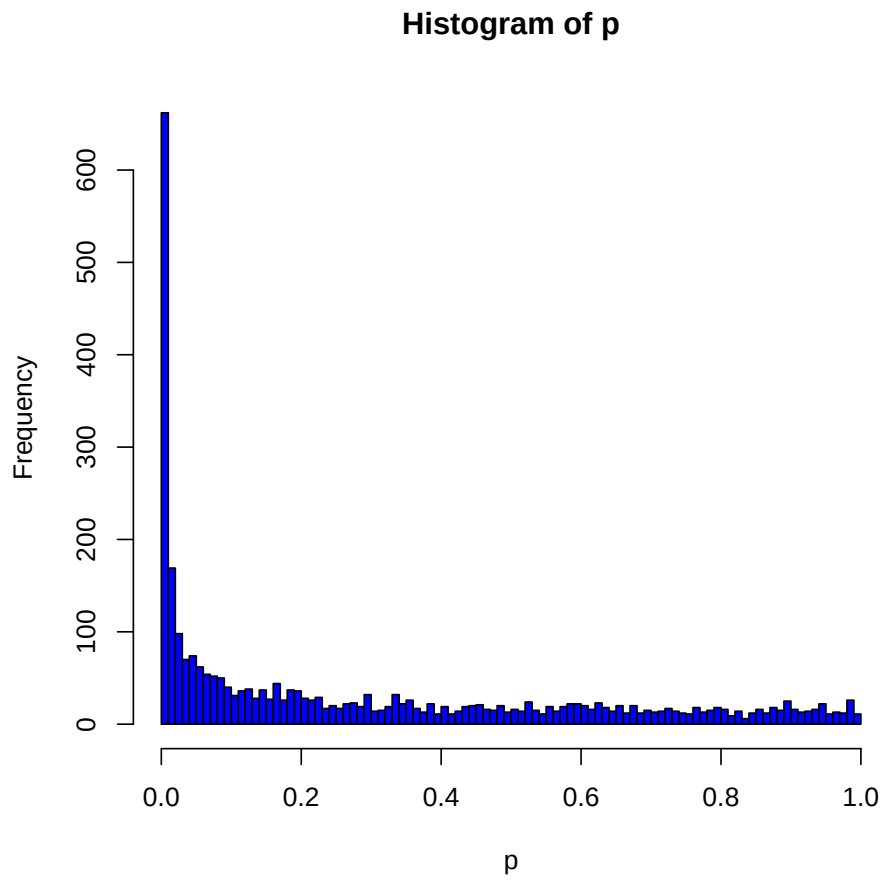
```
> for (i in 1:nrow(dat)) {
+     if (max(dat[i, ]) > 20) {
+         o[i] <- F
+     }
+ }
> dat <- dat[o, ]
> n <- ncol(dat)
> m <- nrow(dat)
> y = golub.cl
> y = y + 1
> ttest <- function(x, y, f = 0) {
+     m1 <- apply(x[, y == 1], 1, mean)
+     m2 <- apply(x[, y == 2], 1, mean)
+     v1 <- apply(x[, y == 1], 1, var)
+     v2 <- apply(x[, y == 2], 1, var)
+     s1 <- sqrt(v1/sum(y == 1))
+     s2 <- sqrt(v2/sum(y == 2))
+     tt <- (m2 - m1)/(sqrt(s1^2 + s2^2) + f)
+     return(tt = tt, ss = (sqrt(s1^2 + s2^2) + f))
+ }
> tt <- ttest(dat, y)$tt
> hist(tt, breaks = 100, col = "red")
```



Let us calculate null statistics and p-values.

```
> tt0 <- 0
> set.seed(123)
> B <- 100
> for (i in 1:B) {
+     v <- sample(y)
+     tt0 <- c(tt0, ttest(dat, v)$tt)
+ }
> tt0 <- tt0[-1]
> att <- abs(tt)
> att0 <- abs(tt0)
> v <- c(rep(T, m), rep(F, m * B))
> v <- v[rev(order(c(att, att0)))]
> u <- 1:length(v)
> w <- 1:m
> p <- (u[v == TRUE] - w)/(B * m)
> p <- p[rank(-att)]
```

```
> hist(p, breaks = 100, col = "blue")
```



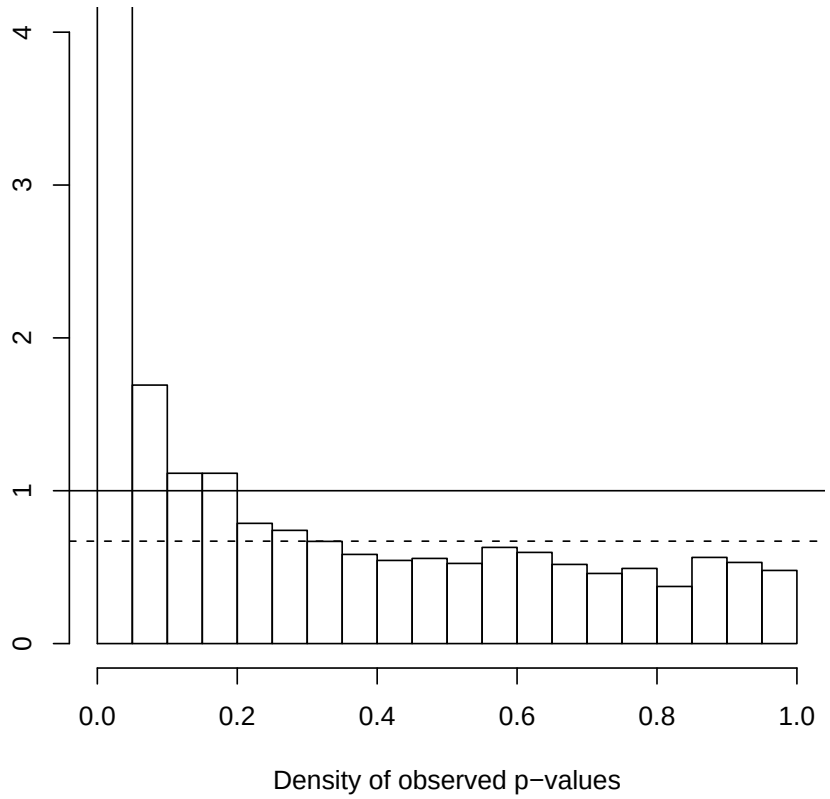
One can adjust the p-values to account for multiple hypothesis testing using the appropriate functions.

```
> qobj <- qvalue(p)
> m1 <- apply(dat[, y == 1], 1, mean)
> m2 <- apply(dat[, y == 2], 1, mean)
> fc <- (m2 - m1)
> qval <- rep(NA, length(o))
> pval <- rep(NA, length(o))
> fchng <- rep(NA, length(o))
> qval[o] <- qobj$q
> pval[o] <- qobj$pval
> fchng[o] <- fc
> for (i in 1:length(o)) {
+     cat(c(round(qval[i], 6), pval[i], round(fchng[i], 3), "\n"),
```

```
+          file = "qvalues.txt", append = T)
+ }
```

We now display the figures as they are given in Storey and Tibshirani's paper (2003). Using a two-sample t-statistic, we calculated a p-value for each of 3051 genes under the null. The figure 1 below displays a density histogram of the 3051 p-values from the Golub et al. data. The dashed line is the density histogram we would expect if all genes were null (not differentially expressed), so it can be seen that many genes are differentially expressed. The dotted line is at the height of the estimate of the proportion of null p-values.

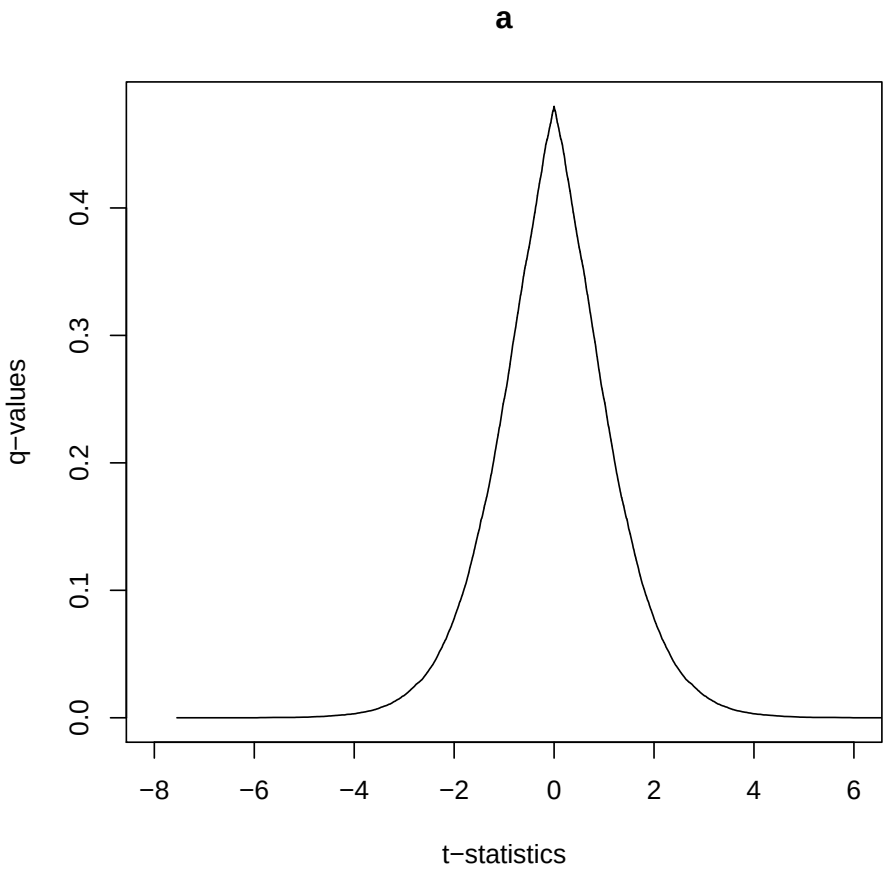
```
> hist(p, nclass = 20, main = " ", xlab = "Density of observed p-values",
+     ylab = " ", freq = F, ylim = c(0, 4))
> abline(h = 1)
> abline(h = 0.67, lty = 2)
```



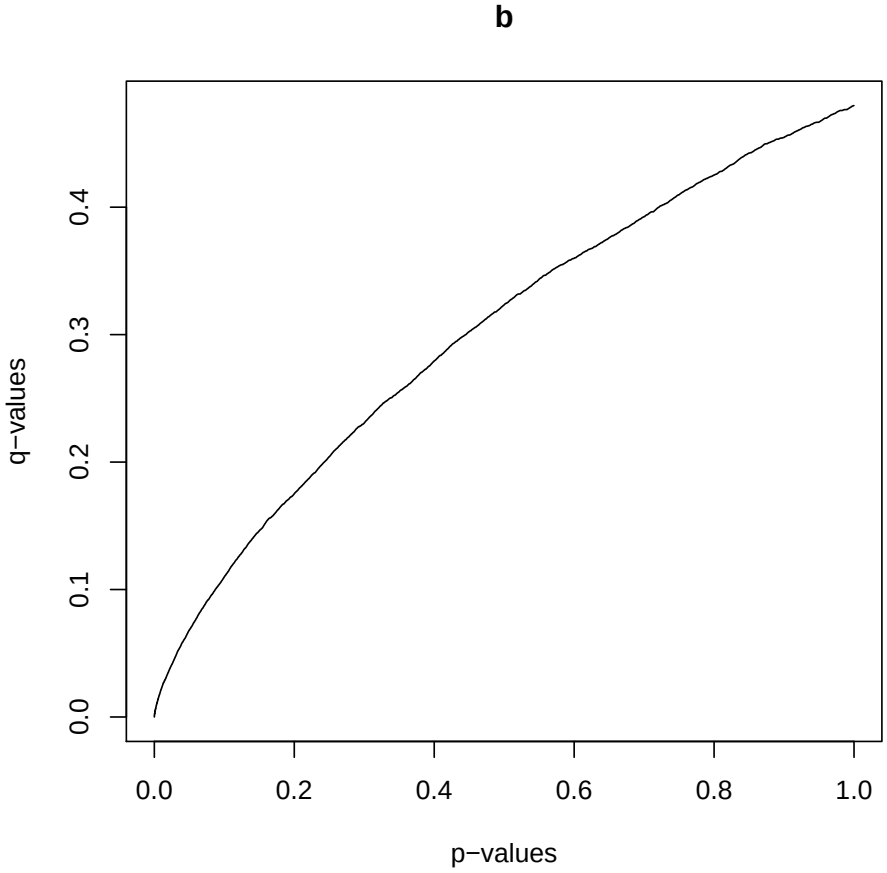
The next four figures display successively (a) The q-values of the genes versus their respective t-statistics. (b) The q-values versus their respective p-values. (c) The number

of genes occurring on the list up through each q-value versus the respective q-value. (d) The expected number of false positive genes versus the total number of significant genes given by the q-values.

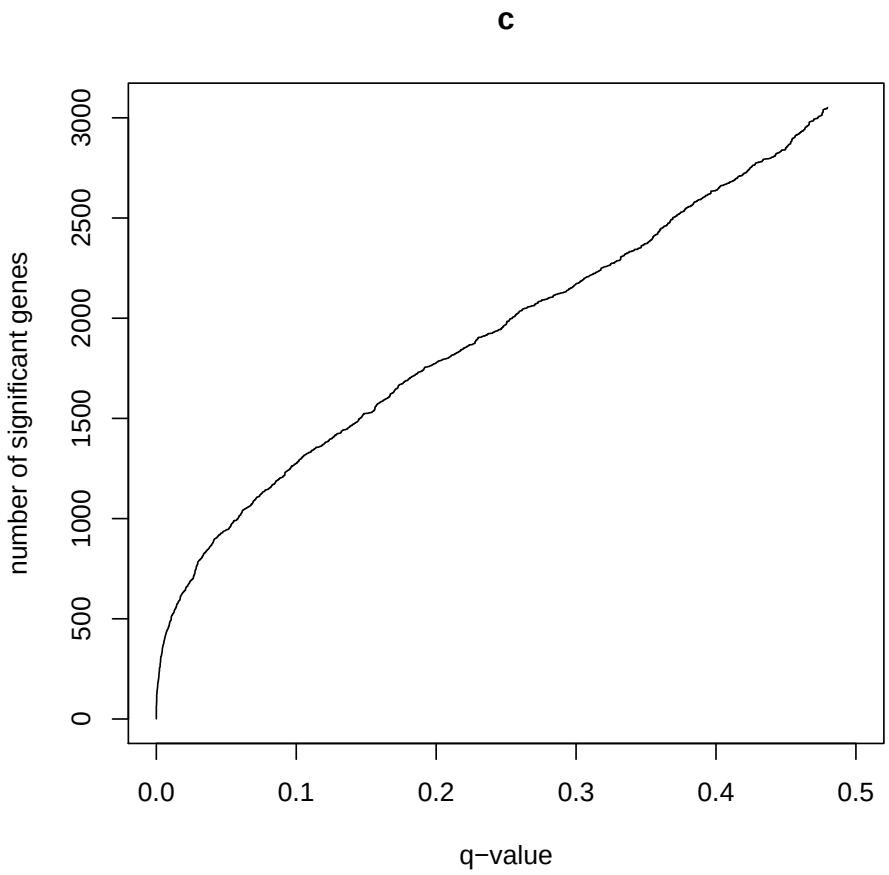
```
> plot(tt[order(tt)], qobj$q[order(tt)], type = "l", xlim = c(-8,
+     6), xlab = "t-statistics", ylab = "q-values", main = "a")
```



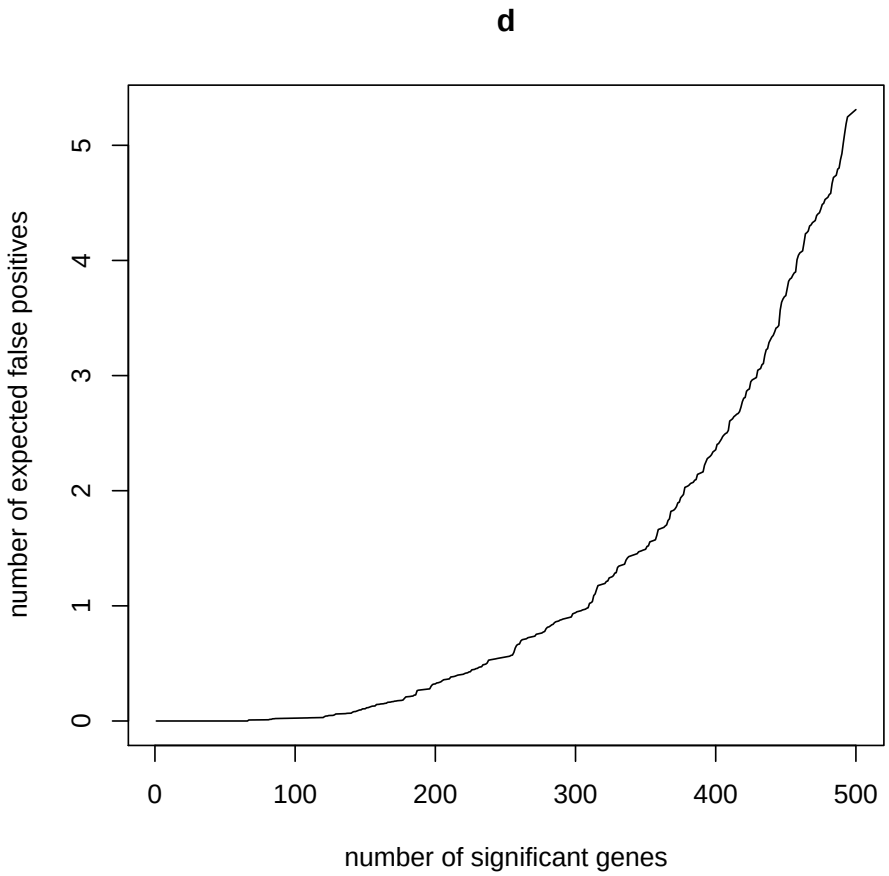

```
> p2 <- qobj$pval[order(qobj$pval)]
> q2 <- qobj$qval[order(qobj$pval)]
> plot(p2, q2, type = "l", xlab = "p-values", ylab = "q-values",
+     main = "b")
```




```
> plot(q2, 1:m, type = "l", xlim = c(0, 0.5), ylim = c(0, 3051),
+     xlab = "q-value", ylab = "number of significant genes",
+     main = "c")
```

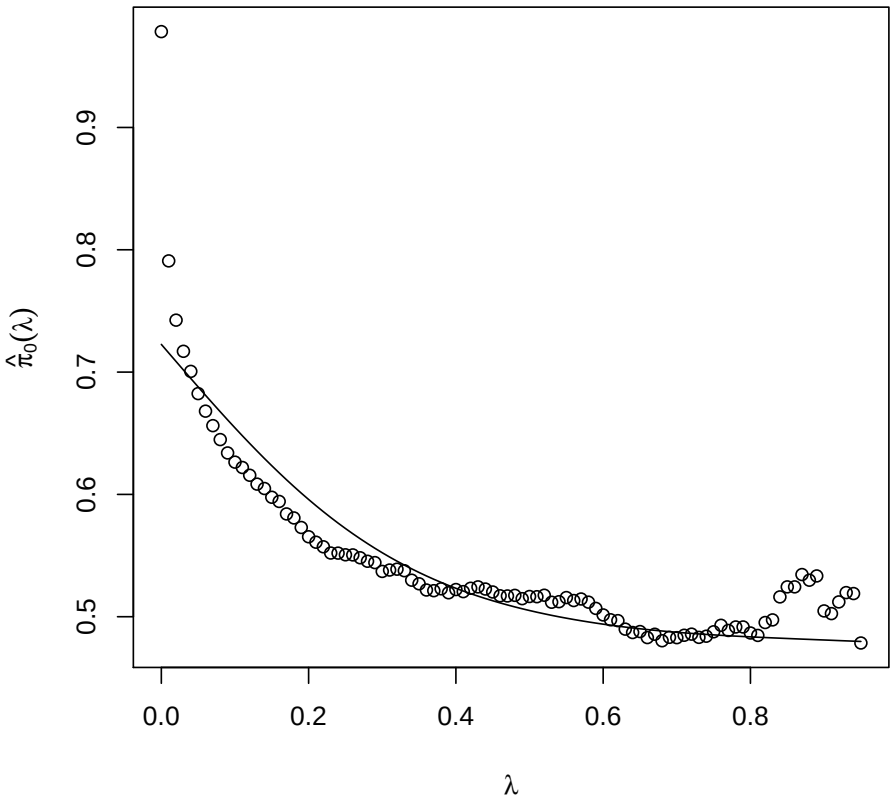



```
> plot(1:500, q2[1:500] * (1:500), type = "l", xlab = "number of significant genes",
+     ylab = "number of expected false positives", main = "d")
```



The next figure displays the estimate $\hat{\pi}_0(\lambda)$ versus λ for the Golub data. The solid line is a natural cubic spline fit to these points to estimate $\hat{\pi}_0(\lambda)$, the proportion of non-differentially expressed genes.

```
> lam <- seq(0, 0.95, 0.01)
> pi0 <- rep(0, length(lam))
> for (i in 1:length(lam)) {
+     pi0[i] <- mean(p > lam[i])/(1 - lam[i])
+ }
> spi0 <- smooth.spline(lam, pi0, df = 3, w = (1 - lam))
> plot(lam, pi0, xlab = expression(lambda), ylab = expression(hat(pi)[0](lambda)))
> lines(spi0)
```



We can easily repeat the whole procedure with the dataset comprising 12 replicates, by simply using `data(esset12)`.