Bioconductor's multtest package

Sandrine Dudoit[1] and Yongchao Ge[2]

August 25, 2004

1. Division of Biostatistics, University of California, Berkeley,
http://www.stat.berkeley.edu/~sandrine
2. Department of Biomathematical Sciences, Mount Sinai School of Medicine, New York,
yongchao.ge@mssm.edu

Contents

1 Overview

The `multtest` package contains a collection of functions for multiple hypothesis testing. These functions can be used to identify differentially expressed genes in microarray experiments, i.e., genes whose expression levels are associated with a response or covariate of interest.

Introduction to multiple testing. This document provides a tutorial for using the `multtest` package. For a detailed introduction to multiple testing consult the document `multtest.intro` in the `inst/doc` directory of the package. See also Shaffer (1995) and Dudoit et al. (2002) for a review of multiple testing procedures and complete references.

Multiple testing procedures implemented in `multtest`. The `multtest` package implements multiple testing procedures for controlling different Type I error rates. It includes procedures for controlling the family–wise Type I error rate (FWER): Bonferroni, Hochberg (1988), Holm (1979),

Sidak, Westfall and Young (1993) minP and maxT procedures. It also includes procedures for controlling the false discovery rate (FDR): Benjamini and Hochberg (1995) and Benjamini and Yekutieli (2001) step–up procedures. These procedures are implemented for tests based on t–statistics, F–statistics, paired t–statistics, block F–statistics, Wilcoxon statistics. The results of the procedures are summarized using adjusted p–values, which reflect for each gene the overall experiment Type I error rate when genes with a smaller p–value are declared differentially expressed. Adjusted p–values may be obtained either from the nominal distribution of the test statistics or by permutation. The permutation algorithm for the maxT and minP procedures is described in Ge et al. (2003).

Help files. As with any R package, detailed information on functions, their arguments and value, can be obtained in the help files. For instance, to view the help file for the function `mt.maxT` in a browser, use `help.start()` followed by `? mt.maxT`.

2 Case study: the ALL/AML leukemia dataset of Golub et al. (1999)

We demonstrate the functionality of this package using gene expression data from the leukemia ALL/AML study of Golub et al. (1999). To load the leukemia dataset, use `data(golub)`, and to view a description of the experiments and data, type `? golub`.

```
> library(multtest, verbose = FALSE)
> data(golub)
```

Golub et al. (1999) were interested in identifying genes that are differentially expressed in patients with two type of leukemias, acute lymphoblastic leukemia (ALL, class 0) and acute myeloid leukemia (AML, class 1). Gene expression levels were measured using Affymetrix high–density oligonucleotide chips containing $p = 6,817$ human genes. The learning set comprises $n = 38$ samples, 27 ALL cases and 11 AML cases (data available at `http://www.genome.wi.mit.edu/MPR`). Following Golub et al. (personal communication, Pablo Tamayo), three preprocessing steps were applied to the normalized matrix of intensity values available on the website: (i) thresholding: floor of 100 and ceiling of 16,000; (ii) filtering: exclusion of genes with $\max / \min \leq 5$ or $(\max - \min) \leq 500$, where max and min refer respectively to the maximum and minimum intensities for a particular gene across mRNA samples; (iii) base 10 logarithmic transformation. Boxplots of the expression levels for each of the 38 samples revealed the need to standardize the expression levels within arrays before combining data across samples. The data were then summarized by a $3,051 \times 38$ matrix $X = (x_{ji})$, where x_{ji} denotes the expression level for gene j in tumor mRNA sample i.

The dataset `golub` contains the gene expression data for the 38 training set tumor mRNA samples and 3,051 genes retained after pre–processing. The dataset includes

- `golub`: a $3,051 \times 38$ matrix of expression levels;
- `golub.gnames`: a $3,051 \times 3$ matrix of gene identifiers;
- `golub.cl`: a vector of tumor class labels (0 for ALL, 1 for AML).

```
> dim(golub)
```

```
[1] 3051   38
```

```
> golub[1:4, 1:4]
```

```
         [,1]     [,2]     [,3]     [,4]
[1,] -1.45769 -1.39420 -1.42779 -1.40715
[2,] -0.75161 -1.26278 -0.09052 -0.99596
[3,]  0.45695 -0.09654  0.90325 -0.07194
[4,]  3.13533  0.21415  2.08754  2.23467
```

```
> dim(golub.gnames)
```

```
[1] 3051    3
```

```
> golub.gnames[1:4, ]
```

```
     [,1] [,2]
[1,] "36" "AFFX-HUMISGF3A/M97935_MA_at (endogenous control)"
[2,] "37" "AFFX-HUMISGF3A/M97935_MB_at (endogenous control)"
[3,] "38" "AFFX-HUMISGF3A/M97935_3_at (endogenous control)"
[4,] "39" "AFFX-HUMRGE/M10098_5_at (endogenous control)"
     [,3]
[1,] "AFFX-HUMISGF3A/M97935_MA_at"
[2,] "AFFX-HUMISGF3A/M97935_MB_at"
[3,] "AFFX-HUMISGF3A/M97935_3_at"
[4,] "AFFX-HUMRGE/M10098_5_at"
```

```
> golub.cl
```

```
 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1
```

3 The `mt.teststat` and `mt.teststat.num.denum` functions

The `mt.teststat` and `mt.teststat.num.denum` functions provide a convenient way to compute test statistics for each row of a data frame, e.g., two–sample Welch t–statistics, Wilcoxon statistics, F–statistics, paired t–statistics, block F–statistics. To compute two–sample t–statistics comparing, for each gene, expression in the ALL cases to expression in the AML cases

```
> teststat <- mt.teststat(golub, golub.cl)
```

The following produces a normal Quantile–Quantile (Q–Q) plot of the test statistics (Figure 1) . In our application, we are not so much interested in testing whether the test statistics follow a particular distribution, but in using the Q–Q plot as a visual aid for identifying genes with "unusual" test statistics. Q–Q plots informally correct for the large number of comparisons and the points which deviate markedly from an otherwise linear relationship are likely to correspond to those genes whose expression levels differ between the control and treatment groups.

```
> postscript("mtQQ.eps")
> qqnorm(teststat)
> qqline(teststat)
> dev.off()

null device
          1

> pdf("mtQQ.pdf")
> qqnorm(teststat)
> qqline(teststat)
> dev.off()

null device
          1
```

We may also wish to look at plots of the numerators and denominators of the test statistics (Figure 2)

```
> tmp <- mt.teststat.num.denum(golub, golub.cl, test = "t")
> num <- tmp$teststat.num
> denum <- tmp$teststat.denum
> postscript("mtNumDen.eps")
> plot(sqrt(denum), num)
> dev.off()

null device
          1

> pdf("mtNumDen.pdf")
> plot(sqrt(denum), num)
> dev.off()

null device
          1
```

4 The `mt.rawp2adjp` function

This function computes adjusted p–values for simple multiple testing procedures from a vector of raw (unadjusted) p–values. The procedures include the Bonferroni, Holm (1979), Hochberg (1988), and Sidak procedures for strong control of the family–wise Type I error rate (FWER), and the Benjamini and Hochberg (1995) and Benjamini and Yekutieli (2001) procedures for (strong) control of the false discovery rate (FDR).

As a first approximation, compute raw nominal two–sided p–values for the $3,051$ test statistics using the standard Gaussian distribution

```
> rawp0 <- 2 * (1 - pnorm(abs(teststat)))
```

Adjusted p–values for these seven multiple testing procedures can be computed as follows and stored in the original gene order in `adjp` using `order(res$index)`

```
> procs <- c("Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD",
+     "BH", "BY")
> res <- mt.rawp2adjp(rawp0, procs)
> adjp <- res$adjp[order(res$index), ]
> round(adjp[1:10, ], 2)
```

```
      rawp Bonferroni Holm Hochberg SidakSS SidakSD   BH BY
 [1,] 0.08          1    1        1       1       1 0.18  1
 [2,] 0.36          1    1        1       1       1 0.54  1
 [3,] 0.92          1    1        1       1       1 0.96  1
 [4,] 0.73          1    1        1       1       1 0.84  1
 [5,] 0.17          1    1        1       1       1 0.32  1
 [6,] 0.21          1    1        1       1       1 0.36  1
 [7,] 0.47          1    1        1       1       1 0.64  1
 [8,] 0.59          1    1        1       1       1 0.74  1
 [9,] 0.99          1    1        1       1       1 0.99  1
[10,] 0.89          1    1        1       1       1 0.94  1
```

5 The `mt.maxT` and `mt.minP` functions

The `mt.maxT` and `mt.minP` functions compute permutation adjusted p–values for the maxT and minP step–down multiple testing procedure described in Westfall and Young (1993). These procedure provide strong control of the FWER and also incorporate the joint dependence structure between the test statistics. There are thus in general less conservative than the standard Bonferroni procedure. The permutation algorithm for the maxT and minP procedures is described in Ge et al. (2003).

Permutation unadjusted p–values and adjusted p–values for the maxT procedure with Welch t–statistics are computed as follows. `mt.maxT` returns p–values sorted in decreasing order of the absolute t–statistics and `order(resT$index)` is used to obtain p–values and test statistics in the original gene order. In practice, the number of permutations B should be several thousands, we set $B = 1,000$ here for illustration purposes.

```
> resT <- mt.maxT(golub, golub.cl, B = 1000)
```

```
b=10        b=20        b=30        b=40        b=50        b=60        b=70        b=80
b=110        b=120        b=130        b=140        b=150        b=160        b=170        b=18
b=210        b=220        b=230        b=240        b=250        b=260        b=270        b=28
b=310        b=320        b=330        b=340        b=350        b=360        b=370        b=38
b=410        b=420        b=430        b=440        b=450        b=460        b=470        b=48
b=510        b=520        b=530        b=540        b=550        b=560        b=570        b=58
```

```
b=610    b=620    b=630    b=640    b=650    b=660    b=670    b=68
b=710    b=720    b=730    b=740    b=750    b=760    b=770    b=78
b=810    b=820    b=830    b=840    b=850    b=860    b=870    b=88
b=910    b=920    b=930    b=940    b=950    b=960    b=970    b=98
```

```
> ord <- order(resT$index)
> rawp <- resT$rawp[ord]
> maxT <- resT$adjp[ord]
> teststat <- resT$teststat[ord]
```

Three functions related to the `mt.maxT` and `mt.minP` functions are `mt.sample.teststat`, `mt.sample.rawp`, and `mt.sample.label`. These functions provide tools to investigate the permutation distribution of test statistics, raw (unadjusted) p–values, and class labels, respectively.

6 The `mt.reject` function

The function `mt.reject` returns the identity and number of rejected hypotheses for several multiple testing procedures and different nominal Type I error rates. The number of hypotheses rejected using unadjusted p–values and maxT p–values for different Type I error rates ($\alpha = 0, 0.1, 0.2, \ldots, 1$) can be obtained by

```
> mt.reject(cbind(rawp, maxT), seq(0, 1, 0.1))$r
```

```
    rawp maxT
0      0    0
0.1 1331  125
0.2 1667  158
0.3 1901  186
0.4 2099  213
0.5 2273  250
0.6 2449  277
0.7 2608  315
0.8 2755  358
0.9 2894  409
1   3051 3051
```

The genes with maxT p–values less than or equal to 0.01 are

```
> which <- mt.reject(cbind(rawp, maxT), 0.01)$which[, 2]
> golub.gnames[which, 2]
```

```
 [1] "ADM Adrenomedullin"
 [2] "CYSTATIN A"
 [3] "Macmarcks"
 [4] "SPTAN1 Spectrin, alpha, non-erythrocytic 1 (alpha-fodrin)"
 [5] "IEF SSP 9502 mRNA"
```

```
 [6] "INDUCED MYELOID LEUKEMIA CELL DIFFERENTIATION PROTEIN MCL1"
 [7] "RB1 Retinoblastoma 1 (including osteosarcoma)"
 [8] "Inducible protein mRNA"
 [9] "LYN V-yes-1 Yamaguchi sarcoma viral related oncogene homolog"
[10] "CD33 CD33 antigen (differentiation antigen)"
[11] "CST3 Cystatin C (amyloid angiopathy and cerebral hemorrhage)"
[12] "FAH Fumarylacetoacetate"
[13] "CTSD Cathepsin D (lysosomal aspartyl protease)"
[14] "IGB Immunoglobulin-associated beta (B29)"
[15] "ACADM Acyl-Coenzyme A dehydrogenase, C-4 to C-12 straight chain"
[16] "CCND3 Cyclin D3"
[17] "HKR-T1"
[18] "Cytoplasmic dynein light chain 1 (hdlc1) mRNA"
[19] "GLUTATHIONE S-TRANSFERASE, MICROSOMAL"
[20] "Leukotriene C4 synthase (LTC4S) gene"
[21] "Putative enterocyte differentiation promoting factor mRNA, partial cds"
[22] "Lysophospholipase homolog (HU-K5) mRNA"
[23] "GB DEF = Homeodomain protein HoxA9 mRNA"
[24] "PLECKSTRIN"
[25] "IRF2 Interferon regulatory factor 2"
[26] "VIL2 Villin 2 (ezrin)"
[27] "ME491  gene extracted from H.sapiens gene for Me491/CD63 antigen"
[28] "RETINOBLASTOMA BINDING PROTEIN P48"
[29] "T-COMPLEX PROTEIN 1, GAMMA SUBUNIT"
[30] "Zyxin"
[31] "LEPR Leptin receptor"
[32] "TOP2B Topoisomerase (DNA) II beta (180kD)"
[33] "C-myb gene extracted from Human (c-myb) gene, complete primary cds, and five complete alt
[34] "APLP2 Amyloid beta (A4) precursor-like protein 2"
[35] "TCRA T cell receptor alpha-chain"
[36] "Interleukin 8 (IL8) gene"
[37] "INTERLEUKIN-8 PRECURSOR"
[38] "MYL1 Myosin light chain (alkali)"
[39] "DHPS Deoxyhypusine synthase"
[40] "MEF2A gene (myocyte-specific enhancer factor 2A, C9 form) extracted from Human myocyte-sp
[41] "X-LINKED HELICASE II"
[42] "LYZ Lysozyme"
[43] "TCF3 Transcription factor 3 (E2A immunoglobulin enhancer binding factors E12/E47)"
[44] "Transcriptional activator hSNF2b"
```

7 The `mt.plot` function

The `mt.plot` function produces a number of graphical summaries for the results of multiple testing procedures and their corresponding adjusted p–values. To produce plots of sorted permutation unadjusted p–values and adjusted p–values for the Bonferroni, maxT, Benjamini and Hochberg

(1995), and Benjamini and Yekutieli (2001) procedures use

```
> res <- mt.rawp2adjp(rawp, c("Bonferroni", "BH", "BY"))
> adjp <- res$adjp[order(res$index), ]
> allp <- cbind(adjp, maxT)
> dimnames(allp)[[2]] <- c(dimnames(adjp)[[2]], "maxT")
> procs <- dimnames(allp)[[2]]
> procs <- procs[c(1, 2, 5, 3, 4)]
> cols <- c(1, 2, 3, 5, 6)
> ltypes <- c(1, 2, 2, 3, 3)
```

For plotting sorted adjusted p–values set the argument `plottype="pvsr"`

```
> postscript("mtpvsr.eps")
> mt.plot(allp[, procs], teststat, plottype = "pvsr", proc = procs,
+     leg = c(2000, 0.4), lty = ltypes, col = cols, lwd = 2)
> dev.off()

null device
          1

> pdf("mtpvsr.pdf")
> mt.plot(allp[, procs], teststat, plottype = "pvsr", proc = procs,
+     leg = c(2000, 0.4), lty = ltypes, col = cols, lwd = 2)
> dev.off()

null device
          1
```

and for plotting adjusted p–values vs. the test statistics use `plottype="pvst"`

```
> postscript("mtpvst.eps")
> mt.plot(allp[, procs], teststat, plottype = "pvst", logscale = TRUE,
+     proc = procs, leg = c(-0.5, 2), pch = ltypes, col = cols)
> dev.off()

null device
          1

> pdf("mtpvst.pdf")
> mt.plot(allp[, procs], teststat, plottype = "pvst", logscale = TRUE,
+     proc = procs, leg = c(-0.5, 2), pch = ltypes, col = cols)
> dev.off()

null device
          1
```

References

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Statist. Soc. B*, 57:289–300, 1995.

Y. Benjamini and D. Yekutieli. The control of the false discovery rate in multiple hypothesis testing under dependency. *The Annals of Statistics*, 29(4):1165–1188, 2001.

S. Dudoit, J. P. Shaffer, and J. C. Boldrick. Multiple hypothesis testing in microarray experiments. *Statistical Science*, to appear, preprint available at UC Berkeley, Division Biostatistics working paper series: 2002-110, `http://www.bepress.com/ucbbiostat/paper110`, 2002.

Y. Ge, S. Dudoit, and T. P. Speed. Resampling-based multiple testing for microarray data analysis. *Test*, to appear, preprint available at the Technical Report #633, Jan. 2003, the Department of Statistics, UC Berkeley, `http://www.stat.berkeley.edu/tech-reports/index.html`, 2003.

T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

Y. Hochberg. A sharper bonferroni procedure for multiple tests of significance. *Biometrika*, 75: 800–802, 1988.

S. Holm. A simple sequentially rejective multiple test procedure. *Scand. J. Statist.*, 6:65–70, 1979.

J. P. Shaffer. Multiple hypothesis testing. *Annu. Rev. Psychol.*, 46:561–584, 1995.

P. H. Westfall and S. S. Young. *Resampling-based multiple testing: Examples and methods for p-value adjustment*. John Wiley & Sons, 1993.

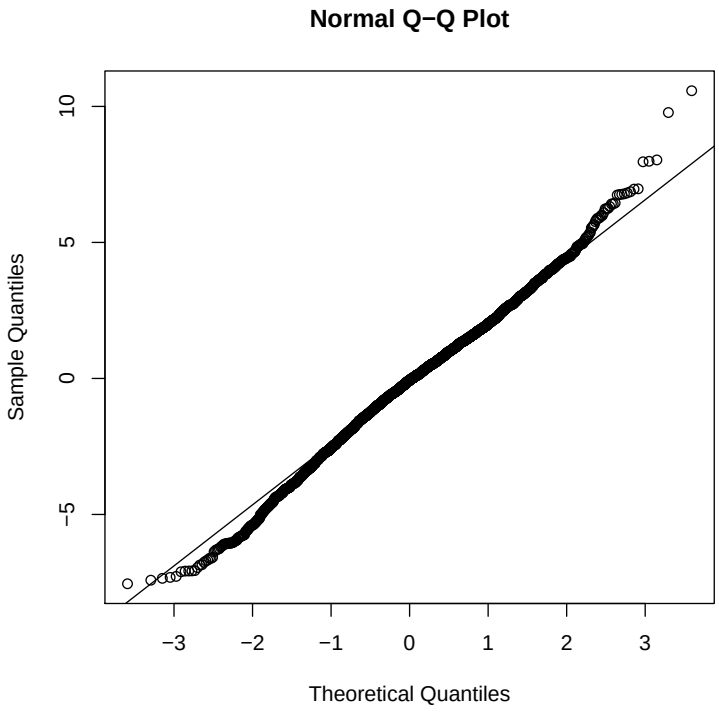

Figure 1: Normal Q–Q plot of t–statistics for leukemia data.

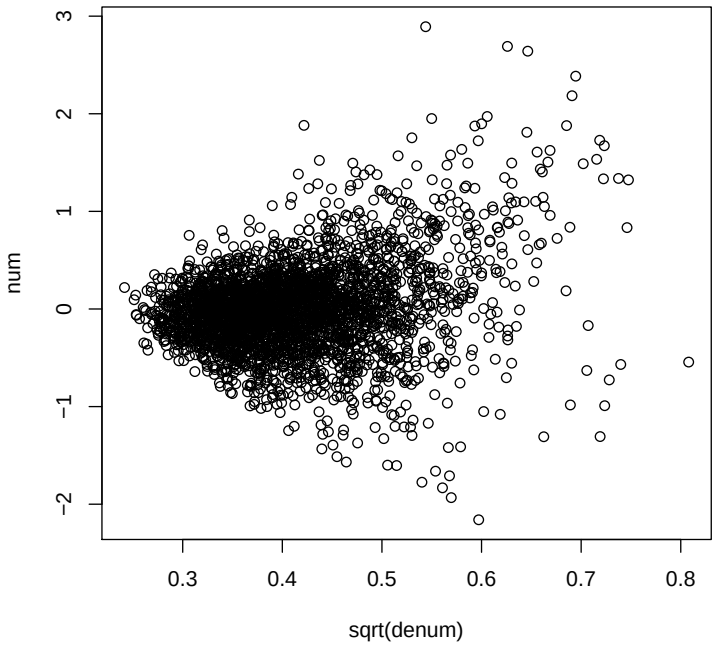

Figure 2: Numerator vs. square root of denominator of the t–statistics for the leukemia data.

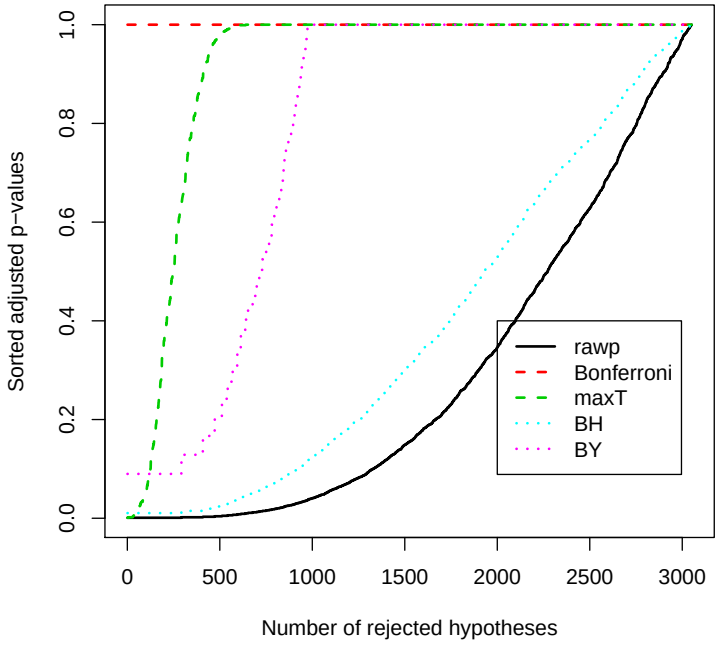

Figure 3: Sorted adjusted p–values for the leukemia data.

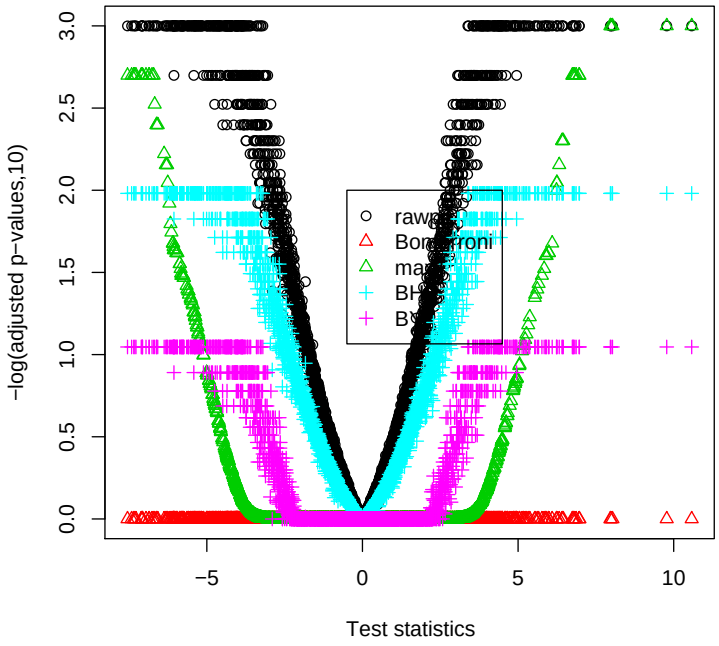

Figure 4: Adjusted p–values (log scale) vs. t–statistics for the leukemia data.