Lab 4: Differential Gene Expression

June 4, 2003

In this lab, we demonstrate how to use R to find genes that are differentially expressed in two populations. We demonstrate two useful plots, the MA-plot and the volcano plot. For a more formal assessment, we use the `multtest` package for obtaining adjusted p-values.

```
> library(Biobase)
> library(ctest)
> library(multtest)
> library(bioclabs)
```

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("eset3")
> genenames <- colnames(pData(eset3))[-1]
```

The concentrations of the sixteen genes in each of six hybridizations are stored in the `phenoData` slot of the `exprSet` object `eset3`.

```
> pData(eset3)
```

```
                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
1521q99hpp_av06          1     1024      0    0.25     0.50      1.0        2
1532m99hpp_av04          0      512   1024    0.00     0.25      0.5        1
1532q99hpp_av04          1     1024      0    0.25     0.50      1.0        2
2353m99hpp_av08          0      512   1024    0.00     0.25      0.5        1
2353q99hpp_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
1521q99hpp_av06        4       8       16       32       64   1024     256
```

```
1532m99hpp_av04        2       4        8       16       32    512     128
1532q99hpp_av04        4       8       16       32       64   1024     256
2353m99hpp_av08        2       4        8       16       32    512     128
2353q99hpp_av08        4       8       16       32       64   1024     256
                1708_at 33818_at 546_at
1521m99hpp_av06     256       32      8
1521q99hpp_av06     512       64     16
1532m99hpp_av04     256       32      8
1532q99hpp_av04     512       64     16
2353m99hpp_av08     256       32      8
2353q99hpp_av08     512       64     16
```

Notice there are two populations and 3 replicates in each. We are interested in identifying genes that are differentially expressed in these two populations.

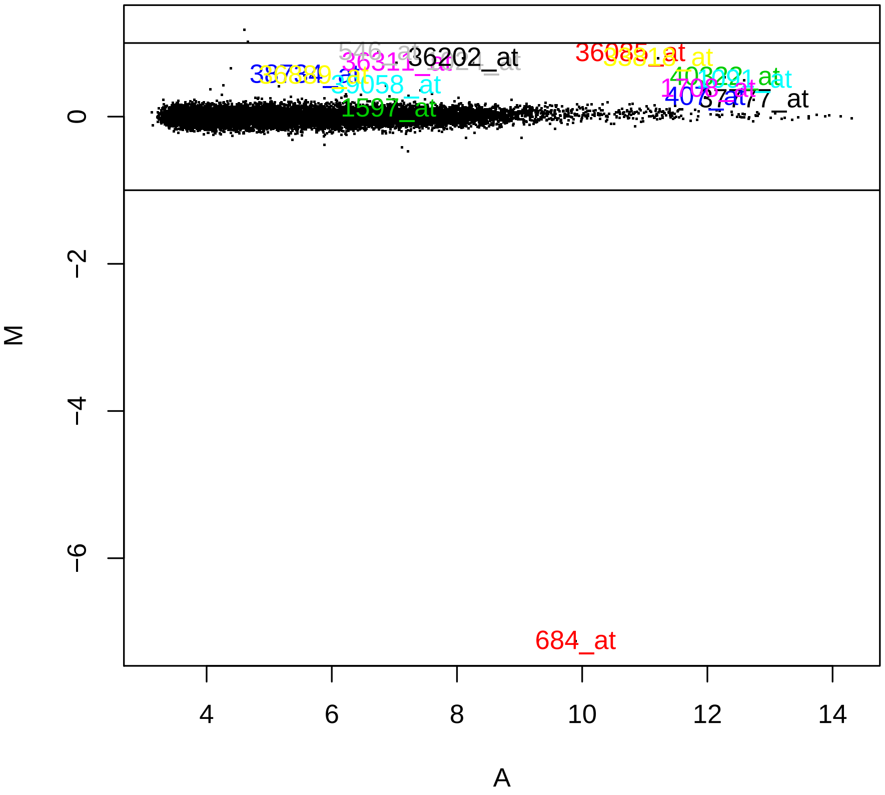
Let us create a matrix containing for each of the 12626 genes on the HGU95a chip (note this really is A and not Av2), its *A-value* or average log intensity, its *M-value* or difference of log intensities (log ratio), its two-sample t-statistic, and its nominal p-value from the t-distribution. The rows 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(eset3$population == 0)
> Index2 <- which(eset3$population == 1)
> scores <- esApply(eset3, 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")
```

The following commands produce an *MA-plot* of the differences of log intensities in the two populations, M, vs. the average log intensities, A.

```
> plot(scores[, 1], scores[, 2], xlab = "A", ylab = "M", pch = ".")
> text(scores[genenames, 1], scores[genenames, 2], genenames, col = 1:16)
> abline(h = c(-1, 1))
```

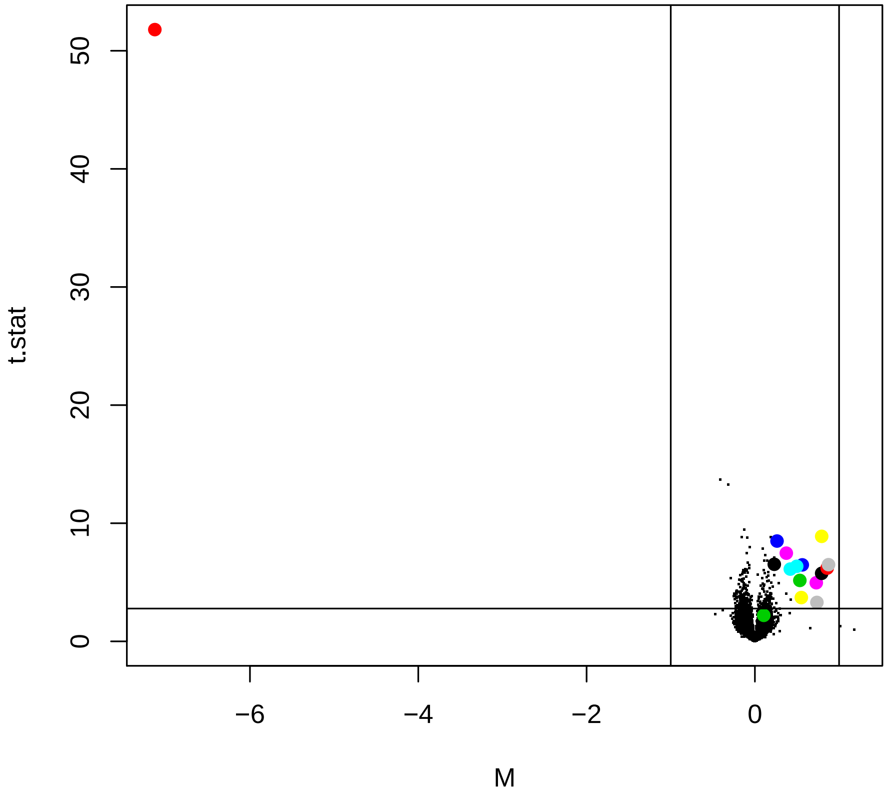


In the MA-plot, points with large vertical deviations (absolute M-values) suggest differentially expressed genes. The horizontal lines show the typical two-fold-change cutoff (because the expression data are on a log 2 scale). The colored symbols correspond to the sixteen genes that are truly differentially expressed, i.e., were spiked in at different concentrations. Other points with large M-values correspond to false positives. One of the sixteen known genes has a low M-value, corresponding to a false negative (`1597_at`). Based on our knowledge of the concentrations for each of the sixteen genes, one expects only one of these genes (`684_at`) to have a negative M-value, i.e., higher expression measure in population 0 than 1 (infinitely more abundant in population 0). This is indeed the point with the very low M-value.

Should we take the variability of the estimates into account? There are only 3 replicates but we can try a t-test. The following is a so-called *volcano plot* of the t-statistic vs. the numerator of the t-statistic, or M-value. Genes in the top left and right corners of the plot correspond to genes with both large absolute differences and large relative (to standard error) differences in expression between the two populations. Although the sixteen known genes tend to have large absolute t-statistics, they standout more in terms of their M-values.

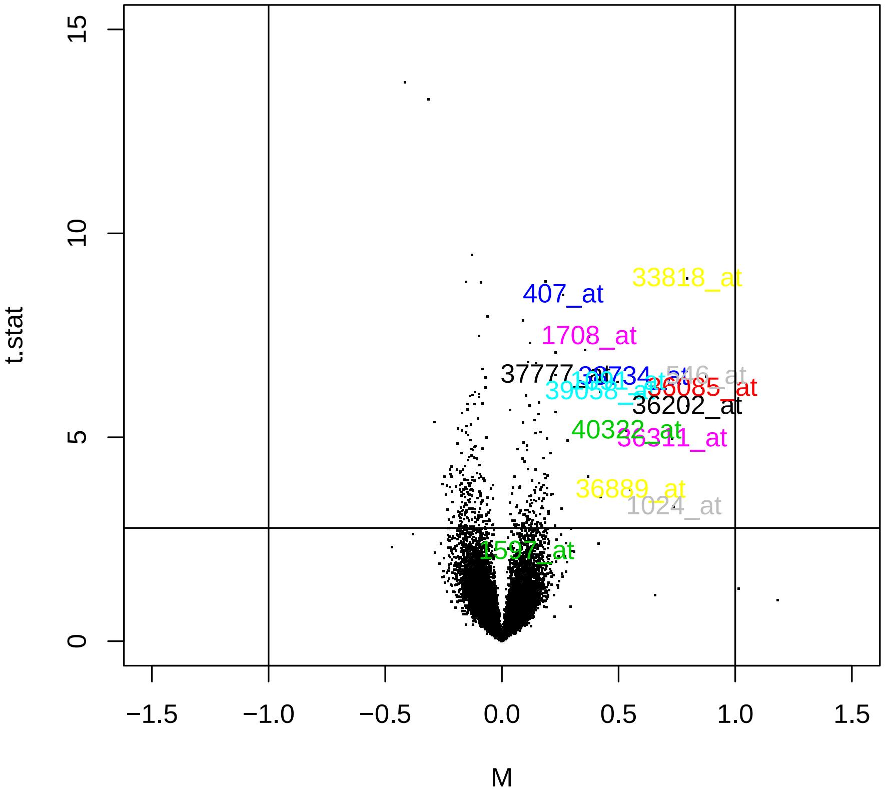
In other situations you may have seen the volcano plot defined as a plot of minus the base 10 logarithm of the data versus fold change. The two versions (ours and this one) are equivalent since all t-statistics are based on the same number of degrees of freedom.

```
> plot(scores[, 2], abs(scores[, 3]), xlab = "M", ylab = "t.stat",
+     pch = ".")
> points(scores[genenames, 2], abs(scores[genenames, 3]), pch = 16,
+     col = 1:16)
> abline(v = c(-1, 1))
> a <- qt(0.975, 4)
> abline(h = a)
```



Note that the gene with small M value really distorts our visual impression of the data. In order to remove that effect we replot the data with new limits. For a close-up, simply change the `xlim` and `ylim`.

```
> plot(scores[, 2], abs(scores[, 3]), xlab = "M", ylab = "t.stat",
+     pch = ".", xlim = c(-1.5, 1.5), ylim = c(0, 15))
> text(scores[genenames, 2], abs(scores[genenames, 3]), genenames,
+     col = 1:16)
> abline(v = c(-1, 1))
> a <- qt(0.975, 4)
> abline(h = a)
```



How many genes have p-values less than 0.05? How about 0.01?

```
> sum(scores[, "p.value"] <= 0.05)

[1] 336

> sum(scores[, "p.value"] <= 0.01)

[1] 74
```

One can adjust the p-values to account for multiple hypothesis testing using the *multtest* package. The function `mt.rawp2adjp` gives adjusted p-values according to various methods using only the raw p-values.

```
> tmp <- mt.rawp2adjp(scores[, 4])
> adj.p.values <- tmp$adjp[order(tmp$index), -1]
> scores <- cbind(scores, adj.p.values)
```

The function maxT computes permutation adjusted *p*-values for the Westfall & Young step-down maxT procedure. One needs to use the original data again. Similarly, for the step-down minP procedure, one can use the function mt.minP.

```
> tmp <- mt.maxT(exprs(eset3), eset3$population)

We'll do complete enumerations

> scores <- cbind(scores, tmp$adjp[order(tmp$index)])
> colnames(scores)[12] <- "maxT"
```

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

```
> round(scores[genenames, ], 2)

             A     M t.stat p.value Bonferroni Holm Hochberg SidakSS SidakSD
37777_at 12.74  0.23   6.53    0.00       1.00 1.00     1.00    1.00    1.00
684_at    9.89 -7.13 -51.79    0.00       0.01 0.01     0.01    0.01    0.01
1597_at   6.91  0.11   2.20    0.09       1.00 1.00     1.00    1.00    1.00
38734_at  5.57  0.56   6.48    0.00       1.00 1.00     1.00    1.00    1.00
39058_at  6.86  0.42   6.12    0.00       1.00 1.00     1.00    1.00    1.00
36311_at  7.04  0.73   4.96    0.01       1.00 1.00     1.00    1.00    1.00
36889_at  5.71  0.55   3.71    0.02       1.00 1.00     1.00    1.00    1.00
1024_at   8.26  0.74   3.30    0.03       1.00 1.00     1.00    1.00    1.00
36202_at  8.10  0.79   5.76    0.00       1.00 1.00     1.00    1.00    1.00
36085_at 10.77  0.86   6.20    0.00       1.00 1.00     1.00    1.00    1.00
40322_at 12.28  0.53   5.16    0.01       1.00 1.00     1.00    1.00    1.00
407_at   11.96  0.26   8.49    0.00       1.00 1.00     1.00    1.00    1.00
1091_at  12.59  0.50   6.36    0.00       1.00 1.00     1.00    1.00    1.00
1708_at  12.01  0.37   7.47    0.00       1.00 1.00     1.00    1.00    1.00
33818_at 11.21  0.79   8.89    0.00       1.00 1.00     1.00    1.00    1.00
546_at    6.75  0.87   6.49    0.00       1.00 1.00     1.00    1.00    1.00
           BH   BY maxT
37777_at 1.00 1.00  1.0
684_at   0.01 0.11  0.2
1597_at  1.00 1.00  1.0
38734_at 1.00 1.00  1.0
39058_at 1.00 1.00  1.0
36311_at 1.00 1.00  1.0
36889_at 1.00 1.00  1.0
```

```
1024_at  1.00 1.00  1.0
36202_at 1.00 1.00  1.0
36085_at 1.00 1.00  1.0
40322_at 1.00 1.00  1.0
407_at   1.00 1.00  1.0
1091_at  1.00 1.00  1.0
1708_at  1.00 1.00  1.0
33818_at 1.00 1.00  1.0
546_at   1.00 1.00  1.0
```

One can make some pretty substantial arguments against this procedure. Make an adjustment for all 12626 probes is probably not appropriate in any situation and is certainly hard to support here. These data arose from a designed experiment where we know which genes are going to change. Not all 12626 were candidates for change. In a real experiment you are likely to find that approximately 40% of the genes are not expressed in the tissue that you are studying. In that case you have corrected for something which should have been excluded from the analysis. There is of course an issue in determining which genes are not differentially expressed is very difficult but the gains from an approximate solution are likely to be quite large.

With 3 replicates we don't expect to have much power. Should we even use t-statistics over the more simple fold change estimates? Let's see which one does better at ranking the sixteen truly differentially expressed genes. Ideally, one would like the sixteen genes to have ranks 1 through 16. It seems like the simple M-value or fold change measure was more successful at identifying the sixteen known genes.

```
> m.ranks <- rank(-abs(scores[, 2]))
> names(m.ranks) <- rownames(scores)
> t.ranks <- rank(-abs(scores[, 3]))
> names(t.ranks) <- rownames(scores)
> cbind(m.ranks, t.ranks)[genenames, ]
```

```
         m.ranks t.ranks
37777_at      67      20
684_at         1       1
1597_at     2068     692
38734_at      11      22
39058_at      17      27
36311_at       9      60
36889_at      12     145
1024_at        8     205
36202_at       7      39
36085_at       5      26
40322_at      13      53
```

```
407_at        37       9
1091_at       14      24
1708_at       21      13
33818_at       6       5
546_at         4      21
```

We can easily repeat the whole procedure with the dataset comprising 12 replicates, by simply using `data(eset12)` instead of `data(eset3)`.