Exploring cDNA data

Course in Practical Microarray Analysis, Heidelberg, March 2003

Wolfgang Huber

The following exercise will show you some possibilities to load data from spotted cDNA microarrays into R, and to explore it using the statistical and visualization facilities of R and Bioconductor.

1.) Preliminaries. To go through this exercise, you need to have installed R>=1.6, recent versions of the Bioconductor libraries Biobase, vsn, multtest, marrayClasses, marrayInput, marrayNorm, and the library lymphoma, which contains the data.

```
> library(vsn)
> library(multtest)
> library(lymphoma)
> library(marrayInput)
> library(marrayNorm)
```

2.) Reading and exploring data files.

a. First, you need to find the directory with the data on your harddisk. The path ends in `library/lymphoma/data`, and this is in the subdirectory where your `R` resides. On the command line, you can use the commands `dir()`, `getwd` and `setwd` to navigate around. In the GUI, you can use *File, Change dir* in the menu.

b. Open the file `lc7b048rex.DAT` in a text editor. This is the typical file format for the results from the image analysis on a cDNA slide. Different image analysis programs use slightly different conventions and column headings, but you can always adapt the input function (see below) to your needs.

c. Use the function `read.delim` to read the file into a *data frame* (that is a rectangular table of data) in R.

```
> x = read.delim("lc7b048rex.DAT")
```

d. The table is too large to print it out as a whole, but we can find out about its size (with the function `dim`) and look at individual rows of the table.

```
> dim(x)
> colnames(x)
> x[1:6, ]
```

3.) Simple plots.

a. Let us first look at the histogram of the values in the column CH1I, that is the channel 1 foreground intensity (see Fig. 1).

```
> hist(x$CH1I)
> hist(log2(x$CH1I), breaks = 100)
> hist(log2(x$CH1I), breaks = seq(5, 15, by = 0.25), col = "blue")
```

b. Now let us look at scatterplots of CHI1 versus CH2I. We can use linear axis scaling or double-logarithmic scaling. There are many options to decorate the plot with your own axis labels and plot title and to use different plot symbols and colors (see Fig. 2).

```
> plot(x$CH1I, x$CH2I)
> plot(x$CH1I, x$CH2I, log = "xy")
> plot(x$CH1I, x$CH2I, log = "xy", main = "lc7b048", xlab = "green",
+     ylab = "red", pch = ".")
> plot(x$CH1I, x$CH1B, log = "xy", xlab = "foreground", ylab = "background",
+     pch = ".")
```

c. Save the plot as PDF, and as Windows metafile. Copy and paste it into an MS-Office application.

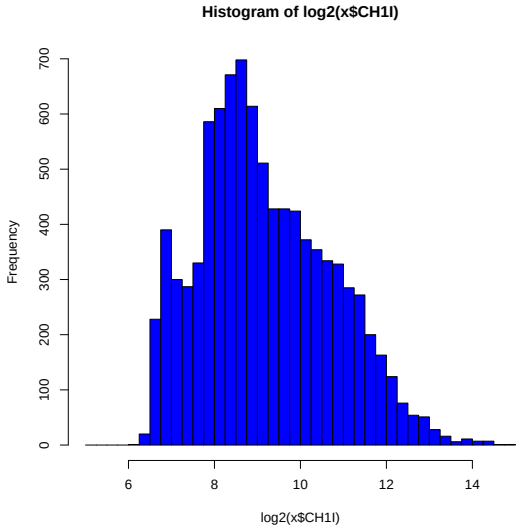

Figure 1:

4.) Calibration and variance stabilization.

a. Subtract the background intensities CH1B, CH2B from the foreground intensities CH1I, CH2I, and store the result in a 9216 x 2 matrix.

```
> y = cbind(x$CH1I - x$CH1B, x$CH2I - x$CH2B)
```

b. What does the function `cbind` do? Use the R online help to find out.

c. Now we can use the function `vsn` to calibrate and transform the data, and plot the result (Fig. 3).

```
> ny = vsn(y)
> plot(ny)
```

d. Have a look at the "vignette" - try the command `openVignette("vsn")`.

a. Reading a collection of files.

b. The file `phenoData.txt` contains information on the samples that were hybridized onto the arrays. Look at it in a text editor. To load it into a `phenoData` object

```
> samples = read.phenoData("phenoData.txt")
> samples
        phenoData object with 3 variables and 8 cases
        varLabels
               sampleid: read from file
               tumotype: read from file
               sex: read from file
> pData(samples)
                sampleid tumotype  sex
lc7b047rex.DAT    CLL-13      CLL    m
lc7b048rex.DAT    CLL-13      CLL    m
lc7b069rex.DAT    CLL-52      CLL    f
lc7b070rex.DAT    CLL-39      CLL    f
lc7b019rex.DAT DLCL-0032     DLCL    f
lc7b056rex.DAT DLCL-0024     DCLC    m
lc7b057rex.DAT DLCL-0029     DLCL    m
lc7b058rex.DAT DLCL-0023     DLCL <NA>
```

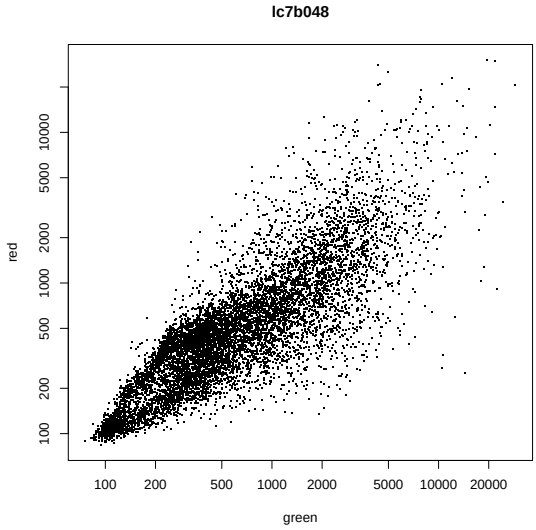

Figure 2:

`phenoData` objects are where the Bioconductor stores information about samples, for example, treatment conditions in a cell line experiment or clinical or histopathological characteristics of tissue biopsies.

c. Now we can load the whole set of 8 slides into the data object `a`.

```
> files = rownames(pData(samples))
> files
> a = read.marrayRaw(files, name.Gf = "CH1I", name.Gb = "CH1B",
+     name.Rf = "CH2I", name.Rb = "CH2B")
```

d. ... and try out different normalization methods:

1. vsn (affine normalization and variance stabilization)
2. maNorm with global median location normalization
3. maNorm with loess for intensity- or A-dependent location normalization using the 'loess' smoother

```
> na1 = vsn(a)
> na2 = maNorm(a, norm = "median", echo = T)
> na3 = maNorm(a, norm = "loess", echo = T)
```

e. These commands take their time! You can save the results into a file with the `save` function, and later restore them with the `load` function. You can use the GUI for the latter.

```
> save(na1, na2, na3, file = "lymphomanorm.rda")
> load(file = "lymphomanorm.rda")
```

f. Now we want to extract the normalized log-ratios. With the present version of Bioconductor, this is still a little bit "manual" - future versions may have a friendlier interface. The first line in the following code creates a three-dimensional array M with space for 9216 genes, 8 samples and 3 different normalization methods. `na1` is the result of `vsn`; the normalized intensities are accessed through the *slot* `h`, using the *accessor* `@`, and the log-ratios are obtained by subtracting the red intensities from the green ones. `na2` and `na3` are the output of `marrayNorm`, and the log-ratios are obtained through the *slot* `maM`.

```
> odd = seq(1, 15, by = 2)
> even = seq(2, 16, by = 2)
> M = array(NA, dim = c(9216, 8, 3))
> M[, , 1] = na1@h[, odd] - na1@h[, even]
```

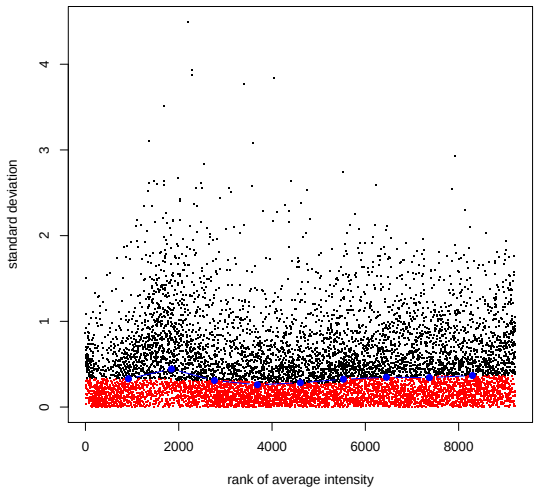

Figure 3:

```
> M[, , 2] = na2@maM
> M[, , 3] = na3@maM
> dim(M)
[1] 9216    8    3
```

5.) Compare the results. Look at scatterplots of the values of M from the same slide, calculated with different normalization methods. Do the values generally agree? How do they differ?

```
> plot(M[, 4, 1], M[, 4, 2], pch = ".", xlab = "vsn", ylab = "loess")
```

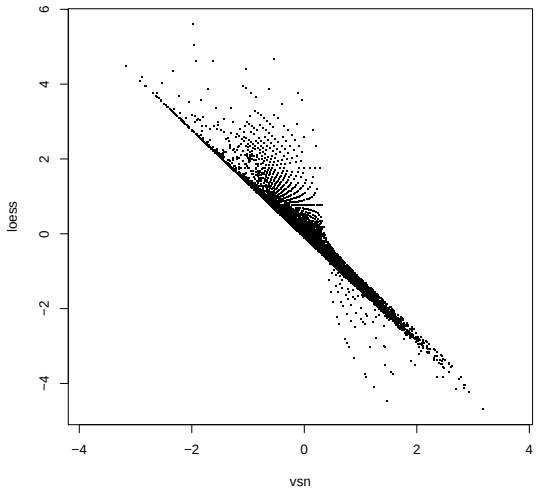

Figure 4:

6.) Testing for differential transcription. Now we are ready to calculate test statistics and to select genes. *Note:* The number of replicates (4 versus 4) that we are considering here is very

small and no significant conclusions about individual genes or individual samples will be derived from that. The full data set contains many more chips. Here we restrict ourselves to a few of them in order to keep things simple for the purpose of this course.

a. Look at the built-in function `t.test`, and at `mt.teststat` from the package `multtest`. Here, we use `mt.teststat` to calculate the t-test statistic for the comparison. The package `multtest` provides extensive functionality to calculate multiple-testing adjustments.

```
> classlabel = c(0, 0, 0, 0, 1, 1, 1, 1)
> t = mt.teststat(M[, , 1], classlabel)
> range(t)
> hist(t, breaks = 100, col = "#fb6090")
```

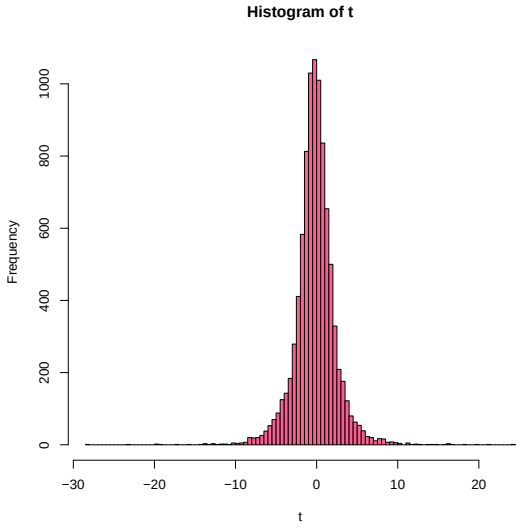

Figure 5:

b. Now we load the spot (gene) description table

```
> spotlabels = read.delim("spotlabels.txt", row.names = 1)
```

print the 5 genes with the lowest values of the t-statistic

```
> selection = order(t)[1:5]
> spotlabels[selection, ]
```

```
     cloneid  genename
4323  235136 GENE 4323
4069   57611 GENE 4069
4331  638320 GENE 4331
2026  823409 GENE 2026
2143  214510 GENE 2143
```

and the 5 genes with the highest values of the t-statistic:

```
> selection = order(t, decreasing = TRUE)[1:5]
> spotlabels[selection, ]
```

```
     cloneid  genename
4532  247003 GENE 4532
8076  128650 GENE 8076
6635  847495 GENE 6635
4586  391898 GENE 4586
739   532891  GENE 739
```