Solutions for chapter Processing Affymetrix Expression Data

Exercise 1

```
> dataPLMx = fitPLM(CLLB)
> boxplot(dataPLM, main="NUSE", ylim = c(0.95, 1.3),
       outline = FALSE, col="lightblue", las=3,
       whisklty=0, staplelty=0)
> Mbox(dataPLM, main="RLE", ylim = c(-0.4, 0.4),
      outline = FALSE, col="mistyrose", las=3,
      whisklty=0, staplelty=0)
```

Exercise 2

There are lots of ways to do that, some of them are listed below.

```
> dim(e)[1]
[1] 12625
> nrow(e)
[1] 12625
> dim(exprs(CLLrma))[1]
[1] 12625
> nrow(CLLrma)
Features
   12625
> length(featureNames(CLLrma))
[1] 12625
```

Exercise 3

```
> par(mfrow=c(1,2))
> myPlot = function(...){
      plot(y = CLLtt$dm, pch = ".", ylim = c(-2,2),
          ylab = "log-ratio", ...)
      abline(h=0, col="blue")
  }
> myPlot(x = a, xlab="average intensity")
> myPlot(x = rank(a), xlab="rank of average intensity")
```

Exercise 4

Plot the two. Perhaps also use an ROC curve?

```
> plot(CLLtt$statistic, CLLeb$t[,2], pch=".")
```

Exercise 5

```
> plot(CLLtt$dm, -log10(CLLeb$p.value[,2]), pch=".",
      xlab="log-ratio", ylab=expression(log[10]~p))
> abline(h=2)
```

Exercise 6

```
> plot(CLLtt$dm, lod, pch=".", xlab="log-ratio",
      ylab=expression(log[10]~p))
> o1 = order(abs(CLLtt$dm), decreasing=TRUE)[1:25]
> points(CLLtt$dm[o1], lod[o1], pch=18, col="blue")
```

Exercise 7

```
> sum(CLLtt$p.value<=0.01)
[1] 243
> sum(CLLeb$p.value[,2]<=0.01)
[1] 261
```

Exercise 8

The values, transformed to a $\log_2$ scale, can be plotted using the code below.

```
> smoothScatter(log2(mms[,1]), log2(pms[,1]),
      xlab="Log2 MM values",
      ylab="Log2 PM values", asp=1)
> abline(a=0, b=1, col="red")
```

Let us look at their relative size.

```
> table(sign(pms-mms))
     -1       0       1
1414590   31828 2993182
```

In a large number of cases, the MM value is larger than the PM value. The simple story of MM measuring non-specific hybridization and PM the sum of non-specific and specific hybridization is hard to hold.

Exercise 9

The two histograms look very different. And we can confirm that, as suggested by the scatterplot in Figure 8, the intensities of the MM probes strongly correlate to those of the PM probes. The histogram for low values is quite skewed, while that corresponding to larger PM values is more symmetric.

```
> grouping = cut(log2(pms)[,1], breaks=c(-Inf, log2(2000),
      Inf), labels=c("Low", "High"))
> multidensity(log2(mms)[,1] ~ grouping, main="", xlab="",
      col=c("red", "blue"), lwd=2)
> legend("topright", levels(grouping), lty=1, lwd=2,
      col=c("red", "blue"))
```

Exercise 10

First, we create a subset `sel` of 500 randomly selected PM probes – this is enough to sample the background correction transformation and reduces the file size of the plots.

```
> sel = sample(unlist(indexProbes(CLLB, "pm")), 500)
> sel = sel[order(exprs(CLLB)[sel, 1])]
```

Then we create the vectors `yo`, `yr` and `yv` with the original, RMA background-corrected and VSN background-corrected intensities for the first array,

```
> yo = exprs(CLLB)[sel,1]
> yr = exprs(bgrma)[sel,1]
> yv = exprs(bgvsn)[sel,1]
```

and plot them. The result is shown in Figure 10.

```
> par(mfrow=c(1,3))
> plot(yo, yr, xlab="Original", ylab="RMA", log="x",
      type="l", asp=1)
> plot(yo, yv, xlab="Original", ylab="VSN", log="x",
      type="l", asp=1)
> plot(yr, yv, xlab="RMA", ylab="VSN", type="l", asp=1)
```

Exercise 11

We need to pay attention to the fact that the non-specific filtering selected different sets of probe sets. In `inboth`, we determine those that are in common.

```
> inboth = intersect(featureNames(CLLvsnf),
      featureNames(CLLf))
> names(CLLvsntt$statistic) = featureNames(CLLvsnf)
> names(CLLtt$statistic) = featureNames(CLLf)
```

```
> plot(CLLtt$statistic[inboth],
       CLLvsntt$statistic[inboth],
       pch=".", xlab="RMA", ylab="VSN", asp=1)
```

The scatterplot is shown in Figure 11.

Exercise 12

We can use the `matplot` function to do this. You should probably either transform the data to the log scale, or use log-scaling in the plot, as we have done. PMs are plotted using a `P`, MMs using a `M`. It is worth noting that for many of the probes, there is no clear separation between the MM values and the PM values (eg probe 1), for others the MM values seem to be higher(!) than the PM values (eg probe 3), and others the PM values are larger than the MM values.

```
> colors = brewer.pal(8, "Dark2")
> Index = indices[["189_s_at"]][seq(along=colors)]
> matplot(t(pms[Index, 1:12]), pch="P", log="y", type="b",
      lty=1, main="189_s_at", xlab="samples",
      ylab=expression(log[2]~Intensity),
      ylim=c(50,2000), col=colors)
> matplot(t(mms[Index, 1:12]), pch="M", log="y", type="b",
      lty=3, add=TRUE, col=colors)
```

The result is shown in Figure 12.

Exercise 13

We can compute the percentage, for each array, by first creating a logical matrix where `TRUE` corresponds to a negative value and `FALSE` corresponds to a non-negative value. Then the column sums of that matrix are the proportions, and if we multiply by 100 we get percentages.

```
> colMeans(newsummary<0)*100
 [1] 20.2 19.6 19.4 18.3 21.0 22.6 21.7 19.6 21.7 21.1
[11] 18.9 18.7 20.6 23.1 19.6 21.0 18.6 21.6 21.4 19.6
[21] 19.7 19.8
```