Introduction to the `maCorrPlot` package

Alexander Ploner
Medical Epidemiology & Biostatistics
Karolinska Institutet, Stockholm
email: `alexander.ploner@ki.se`

October 29, 2024

Abstract

The `maCorrPlot` package implements the graphical diagnostic for microarray normalization described in Ploner et al. [2005].

Contents

1 Preparations

Load the library:

```
> library(maCorrPlot)
```

The example data comprises two data sets A and B, both with MAS5 and RMA expression values as well as MAS5 absent/present calls:

```
> data(oligodata)
> ls()
```

```
[1] "datA.amp"  "datA.mas5" "datA.rma"  "datB.amp"  "datB.mas5"
[6] "datB.rma"
```

All example data has the same size, 5000 genes and 50 samples, and is anonymized. The MAS5 values are logarithmized and normalized to the global mean:

```
> dim(datA.mas5)

[1] 1000   30

> datA.mas5[1:5, 1:5]

       patA1    patA2     patA3    patA4    patA5
g1  1.892772 1.777989 2.2940157 2.155852 1.646282
g34 4.545046 5.265904 4.5486268 4.084411 4.737238
g46 5.028556 5.723734 5.2496368 5.716545 6.326813
g50 1.828328 1.450986 0.8753628 1.506495 1.341759
g83 5.023097 5.300183 5.4978311 5.634437 5.901166
```

2 A basic run

The idea is that random pairs of genes should *on average* be have zero correlation. We start therefore by drawing a random sample of pairs from the RMA values for data set A:

```
> corrA.rma = CorrSample(datA.rma, np=1000, seed=213)
```

This specifies the data set to use, the number of random pairs of genes (1000), and the starting seed for the random sampling (for the sake of replicability).

The random sample can then be plotted to produce Figure 1:

```
> plot(corrA.rma)
```

This plots the average correlation as a function of the average variability of the pair of genes: the average correlation for the pairs of genes with the lowest variability is clearly positive, and the confidence intervals (vertical bars) indicate that this effect is much larger than expected by chance. We conclude that there is some issue with the normalization of low-variance genes in the RMA data.

Note that `corrA.rma` is basically just a data frame that contains the required information for plotting, plus some auxiliary quantities:

```
> corrA.rma[1:5,]

    Correlation     StdDev     Mean       sd1       sd2       m1
1 -0.0849230177 0.13055150 7.325513 0.1976305 0.6605836 7.556292
2  0.0008616332 0.10759448 8.358497 0.3315128 0.3245561 9.262462
3  0.0435180188 0.23726543 7.391163 0.3314405 0.7158613 8.038535
4  0.1435507207 0.06836852 8.120889 0.2169918 0.3150743 7.168125
5  0.1415609560 0.17410383 6.543966 0.9446680 0.1843016 6.402677
        m2 ndx1 ndx2
1 7.094733  440  428
```

```
> jj = plot(corrA.rma)
> print(jj)
```

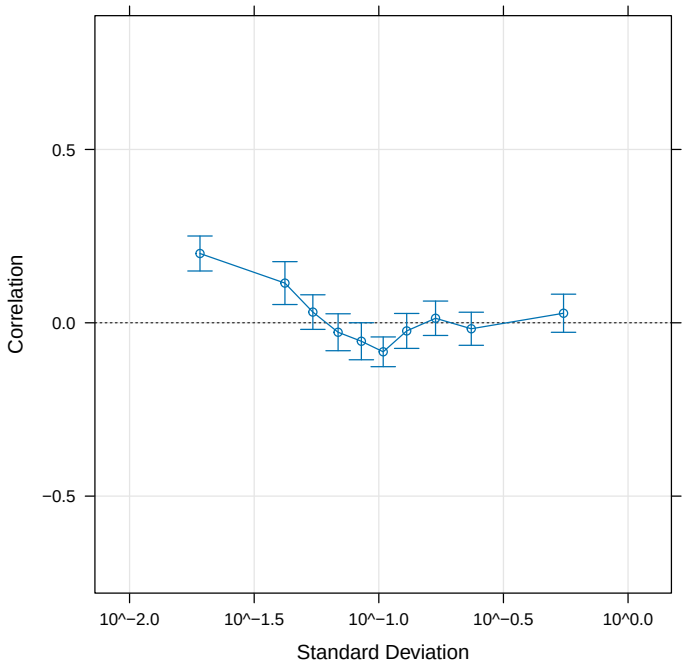

Figure 1: Basic correlation plot for the RMA values of data set A. Note that the assignment to `jj` and the extra print command shown above the figure are required by the vignette setup; `plot(corrA.rma)` on its own is sufficient for interactive data analysis.

```
2 7.454533  272  329
3 6.743790  957  737
4 9.073653  727  733
5 6.685255  546  144
```

3 Extra plotting options

By default, only the average correlations are shown, as in Figure 1, but the underlying correlations and average standard deviations can be included via `scatter=TRUE`, see Figure 2.

Additionally, a very simple theoretical model for lack of normalization can

```
> jj = plot(corrA.rma, scatter=TRUE, curve=TRUE)
> print(jj)
```

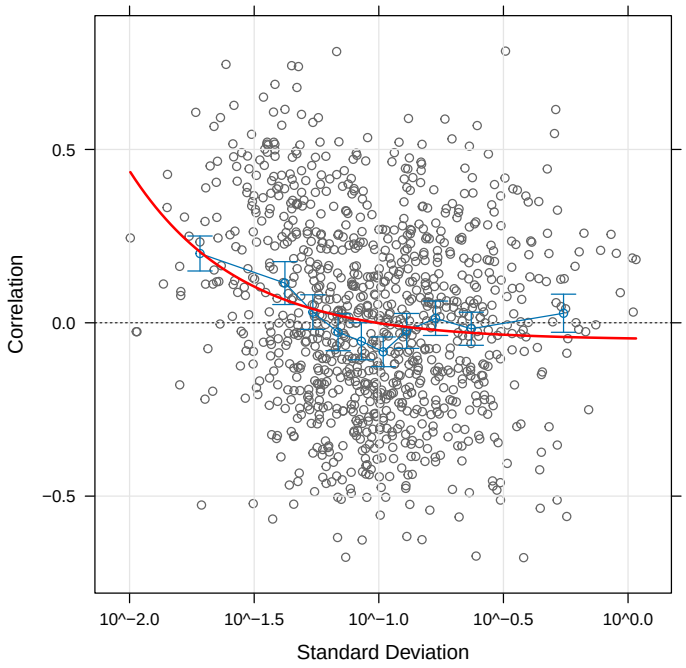

Figure 2: Correlation plot for the RMA values of data set A, with added scatter and model curve

be overlayed on the curve of average correlations via `model=TRUE`, see Ploner et al. [2005] for details. In Figure 2, the agreement betwen the red model curve and the average correlations is only approximate, indicating that the situation here is more complex than the simple model.

4 Multiple plots

4.1 Comparing datasets and/or expression measures

Multiple objects created by `CorrSample` can be shown in the same plot. This allows easy comparison of normalization problems between different expression measures, different data sets, or both.

Let's start with comparing the MAS5 expression values of dataset A with the RMA values. First we draw a random sample of genes from dataset A and

compute their correlations etc.:

```
> corrA.mas5 = CorrSample(datA.mas5, np=1000, seed=213)
```

Then we plot the sampled correlations for both data sets:

```
> plot(corrA.mas5, corrA.rma, cond=c("MAS5","RMA"))
```

This produces Figure 3. Note that we use the argument `cond` to specify appropriate labels for the plot; if it is missing, the plots are just numbered in the order in which they appear in the argument list.

Let's assume that we want to compare MAS5 and RMA for both datasets, A and B, at the same time. We first sample random pairs of genes for dataset B:

```
> corrB.rma = CorrSample(datB.rma, np=1000, seed=214)
> corrB.mas5 = CorrSample(datB.mas5, np=1000, seed=214)
```

We can then go ahead and plot everything in the same manner as before:

```
> plot(corrA.mas5, corrA.rma, corrB.mas5, corrB.rma, cond=c("MAS5/A","RMA/A","MAS5/B","RMA/B
```

This perfectly feasible, though `cond` can be specified differently to create the somewhat more pleasing arrangement shown in Figure 4:

```
> plot(corrA.mas5, corrA.rma, corrB.mas5, corrB.rma, cond=list(c("MAS5","RMA","MAS5","RMA"),
```

Here we specify a list of two factors that characterize the objects to be plotted independently, which results in the nested plot titels in Figure 4.

4.2 Comparing different groups of genes

We may want to compare directly how well different groups of genes are normalized within the same data set and for the same expression measure. One possibility of doing this would be to sub-divide the original dataset, to run `CorrSample` separately on the different subsamples, and to plot these together, as demonstrated in the previous section.

The `plot`-method for objects created by `CorrSample` however supports a different mechanism that allows to show different correlation curves overlayed in the same plot. We demonstrate this mechanism by comparing the correlation curves of genes that were reliably measured (i.e. have a high number of present calls) and genes that were not (i.e. have a low percentage of present calls).

We start by computing the number of present calls for each gene:

```
> pcntA = rowSums(datA.amp=="P")
```

Each gene has between 0 and 30 present calls; in our example, genes tend to have either all present calls or no present calls, see Figure 5(a). Correspondingly, if we compute the average number of present calls for the random pairs of genes for which we have computed the correlations,

```
> jj = plot(corrA.mas5, corrA.rma, cond=c("MAS5","RMA"))
> print(jj)
```

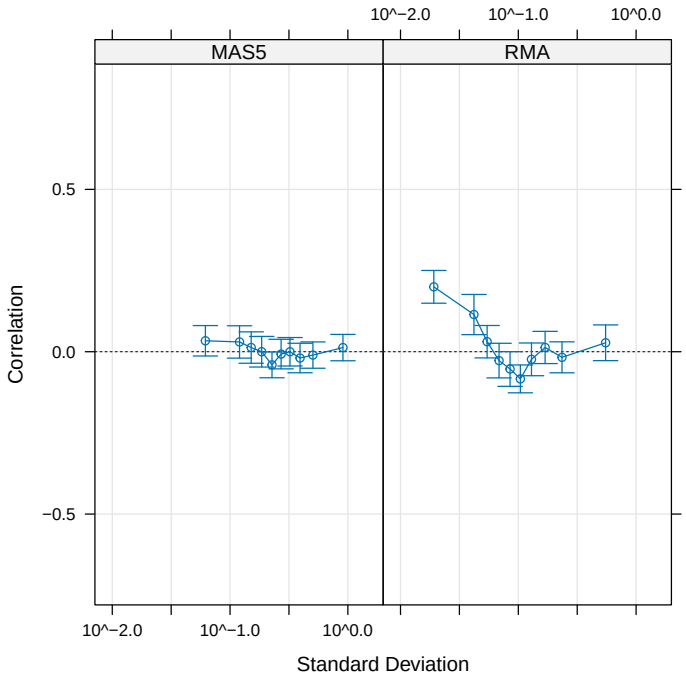

Figure 3: Multiple correlation plot, showing the correlation plot for both MAS5 and RMA values of dataset A.

```
> jj = plot(corrA.mas5, corrA.rma, corrB.mas5, corrB.rma, cond=list(c("MAS5","RMA","MAS5","R
> print(jj)
```

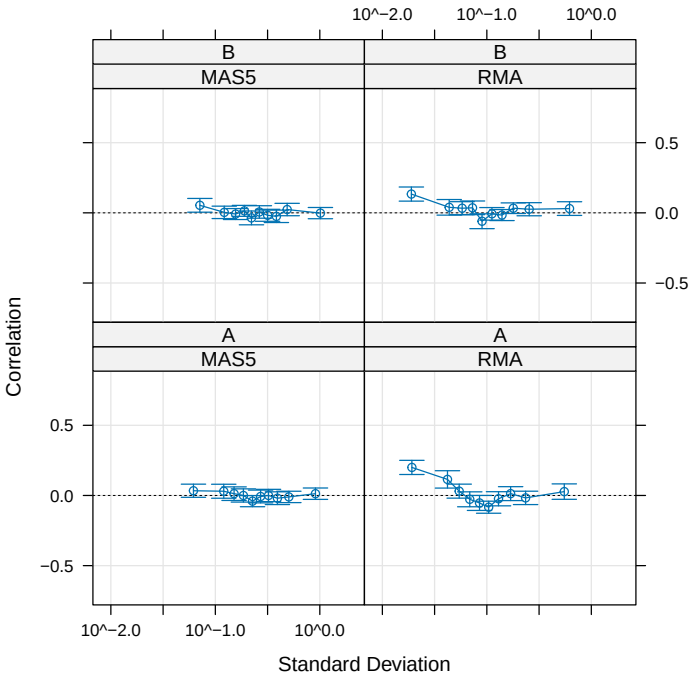

Figure 4: Multiple correlation plot, showing the correlation plot for MAS5 and RMA values for both dataset A and B.

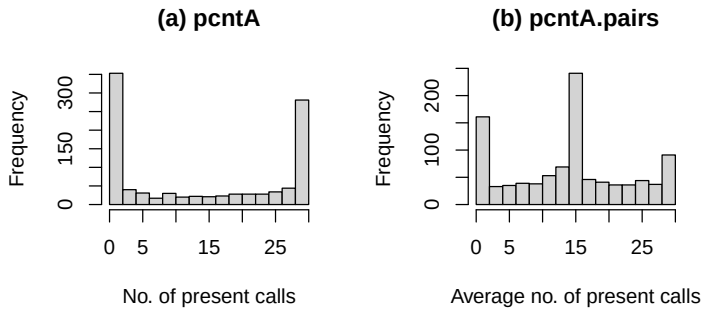

Figure 5: Distribution of the number of present calls on the 30 chips for all 5000 genes in dataset A (left), and of the average number of present calls for the random pairs of genes oused in `corrA.rma` (right).

```
> pcntA.pairs = (pcntA[corrA.rma$ndx1]+pcntA[corrA.rma$ndx2])/2
```

these pairs will come in three flavors: pairs that are both present, pairs that are both absent, and mixed pairs, see Figure 5(b), and we can define a useful classification of the pairs of genes as follows:

```
> pgrpA.pairs = cut(pcntA.pairs, c(0, 10, 20, 30), include.lowest=TRUE)
> table(pgrpA.pairs)

pgrpA.pairs
 [0,10] (10,20] (20,30]
    306     450     244
```

This classification is now simply passed to `plot` via the argument `group`:

```
> plot(corrA.rma, groups=pgrpA.pairs, auto.key=TRUE)
```

This results in Figure 6, showing the correlation cruves for the three different classes of pairs in different colors. It appears that the extra correlation at the origin is due to pairs where both genes have few or no present calls.

Note that the argument `auto.key` creates the legend at the top of Figure 6. Generally, all arguments to `xyplot` can be used in calls to this `plot`-method.

4.3 Multiple expression measures and multiple groups combined

Figure 7 shows how multiple expression measures and groups of genes can be combined in the same plot. Note that we make use of the fact that we had the same random seed, and therefore the same random pairs of genes for both MAS5 and RMA, therefore we can recycle the classification variable `pgrpA.pairs`.

```
> jj = plot(corrA.rma, groups=pgrpA.pairs, auto.key=TRUE)
> print(jj)
```

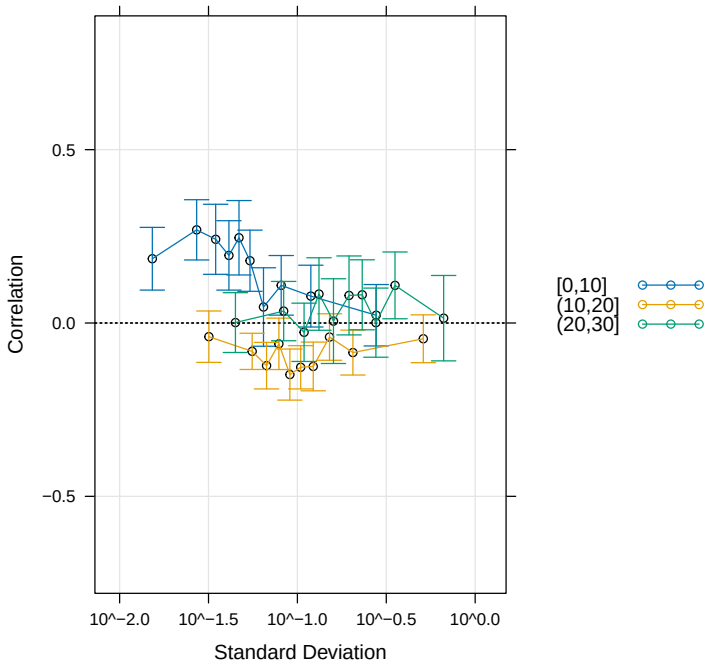

Figure 6: Multiple correlation plots for pairs of genes with low, medium, and high average number of present calls between them.

```
> jj = plot(corrA.mas5, corrA.rma, cond=c("MAS5","RMA"), groups=list(pgrpA.pairs, pgrpA.pair
> print(jj)
```

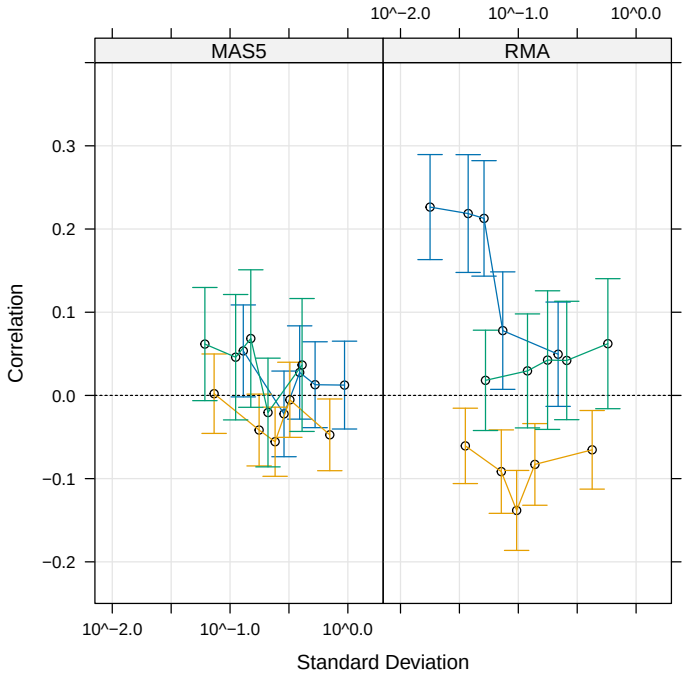

Figure 7: Multiple correlation plot for different groups of genes and two different expression measures. The number of intervals for which we compute average correlations has been reduced to five to avoid clutter, and the vertical axis of the plot has been reduced to avoid empty space.

5 Caveats

These plots are a fairly useful tool for detecting spurious correlations due to failure of normalization, but please consider the following points:

1. This approach has been developed for oligonucleotide/one-dye chips. The principle should apply equally to cDNA/two-dye chips, and some people have found it useful in that context, but I've no experience with it myself.

2. This approach will work for chips with a large number of genes, say an Affymetrix HGU133A chip. For much smaller chips, especially boutique chips, the assumption of zero average correlation may not make sense.

3. This approach can be used to compare different normalizations or expression measures for a specific data set, but it has nothing to say about the quality of an expression measure in its own right, e.g. its bias or lack thereof in estimating fold changes, or the like.

References

Ploner A, Miller LD, Hall P, Bergh J and Pawitan Y. (2005) Correlation test to assess low-level processing of high-density oligonucleotide microarray data. *BMC Bioinformatics*, 6:80.