Bioconductor's nnNorm package

Laurentiu A. Tarca[1]

May 31, 2006

Wayne State University, Medicine - Perinatology Research Branch, Detroit, Michigan, 48201.
`http://vortex.cs.wayne.edu/tarca`

1 Overview

The `nnNorm` package contains mainly a function for intensity and spatial normalization of cDNA two color data, or paired single channel data, based on neural networks fitting. Functionality to compare the distributions of the normalized log ratios is also provided. For the simpler case when only intensity normalization is desired (univariate distortion color model), we provide functionality to plot the bias estimate against the level of intensity for every print tip group.

Introduction to intensity and spatial normalization. This document provides a tutorial for using the `nnNorm` package. For a detailed description of the principles and algorithmic implemented by this package consult Tarca et al. (2005).

Spatial and intensity normalization implemented in `nnNorm`. The `nnNorm` package implements intensity and spatial normalization of cDNA data based on neural networks. In this approach the log average intensity `A` as well as the spatial coordinates `X` and `Y` (obtained by binning the print tip group space) are used as regressors into a neural network model. Resistance to outliers is enhanced by assigning to each spot a weight which is dependent on how extreme its log ratio, `M`, is with respect to that of the spots having about the same `A` level belonging to the same print tip group. A 5-fold crossvalidation procedure is used to obtain the bias estimate for every spot.

2 Changes with respect to previous versions

The first version of `nnNorm` was 1.4.0 and released within bioC 1.7. Thank to the users' feed-back, `nnNorm` continued to improve through the subsequent versions. We would like to thank especially to: Ken Kompass (Wash Univ, St Louis, MO, USA) for pointing out the need of a verbose argument to the maNormNN function; Elizabeth Thomas (CHR, Harvard University) and Philippe Hupé (Institut Curie, Paris France) who provided us with data sets on which we could test new methods to detect convergence problems in neural networks training in the first version of nnNorm, 1.4.0. The changes in the actual `nnNorm` version with respect to the previous ones include: a) avoiding convergence problems in neural networks fitting, b) using a 5-fold cross-validation method instead of 4-fold used previously, c) taking an average over the predictions of several models trained on the same data but starting with different initial weights d) adding a verbose parameter and e) slightly different parameter values of the algorithm to increase even more the robustness.

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 `maNormNN` in a browser, use `help.start()` followed by `? maNormNN`.

3 Case study: Apo AI dataset, Callow et al.

We demonstrate the functionality of this package using gene expression data from the Apo AI study of Callow et al. (2000). To load the Apo dataset in a object called `Apo` of type `marrayRaw`, we use the following lines:

```
> library(marray)
> dataweb <- "http://www.stat.berkeley.edu/users/terry/zarray/Data/ApoA1/rg_a1ko_morph.txt"
> data <- read.table(dataweb, header = TRUE, sep = ",", dec = ".")
> ApoLayout <- new("marrayLayout", maNgr = 4, maNgc = 4, maNsr = 19,
+     maNsc = 21)
> spots <- 4 * 4 * 19 * 21
> Gb <- Rb <- array(c(0), c(spots, 16))
> Gf <- as.matrix(data[, seq(2, 33, 2)])
> Rf <- as.matrix(data[, seq(3, 33, 2)])
> labs <- c(paste(c("c"), 1:8, sep = ""), paste(c("k"), 1:8, sep = ""))
> maInfo <- new("marrayInfo", maLabels = labs, maInfo = data.frame(Names = labs))
> Apo <- new("marrayRaw", maRf = Rf, maGf = Gf, maRb = Rb, maGb = Gb,
+     maLayout = ApoLayout, maTargets = maInfo)
```

Note that the background values were set to zero as the data file contains background corrected red and green intensities.
Due to several factors, the bias in the microarray experiments may depend not only on the average log intensity level but also on the spatial coordinates Yang et al. (2002). Figure 1 shows the reconstructed image of log ratios `M`, for slide No. 16 in the Apo AI data set. This image is obtained using the `maImage` function of `marray` package.

```
> RGcol <- maPalette(low = "green", high = "red", k = 20)
> maImage(Apo[, 16], x = "maM", col = RGcol, zlim = c(-2, 2))
```

Observe the uneven distribution of M values in the print tip groups situated at the bottom of the figure.

Now we perform normalization with the method of the `nnNorm` package called `maNormNN`. This function returns a `marrayNorm` type object (containing the normalized log ratios). The `maNormNN` function is a print-tip oriented normalization method.

```
> library(nnNorm)
> ApNN2DA <- maNormNN(Apo)
```

Figure 2 shows the log ratios, for the same data as in Figure 1 after intensity and spatial normalization. Observe a rather uniform distribution of red and green colored spots.

```
> maImage(ApNN2DA[, 16], x = "maM", col = RGcol, zlim = c(-2, 2))
```

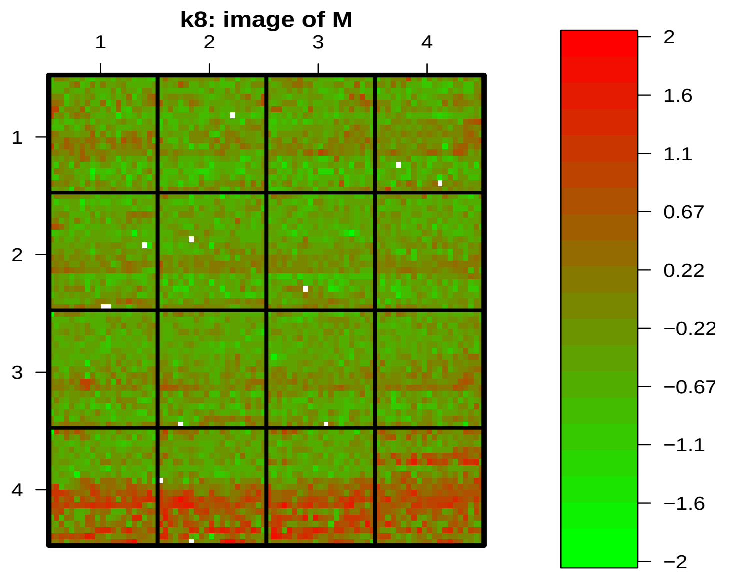


Figure 1: Image of raw M values for Apo data set, slide No. 16.

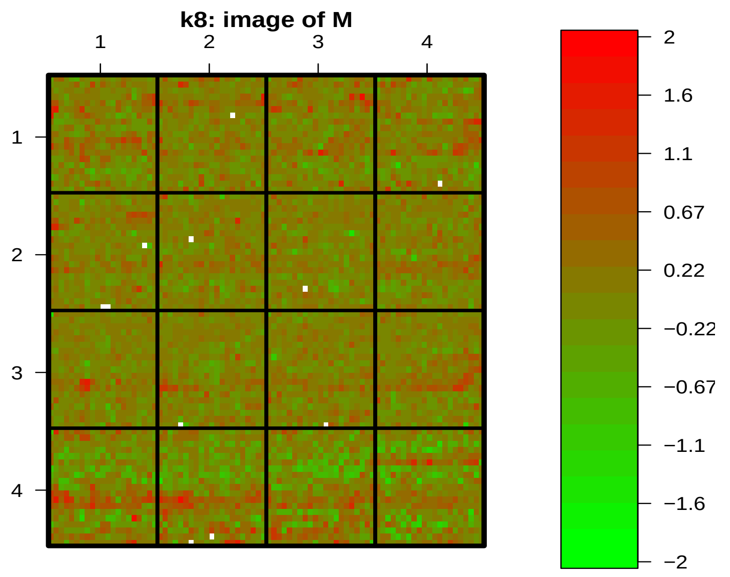


Figure 2: Image of M values after nnNorm normalization, for Apo data set, slide No. 16.

We may compare the robust neural networks normalization method with other existing approaches that takes into account intensity and spatial bias fluctuations, like for e.g. a composite normalization method available in `marray` package.

```
> AcPLo2D <- maNormMain(Apo, f.loc = list(maNorm2D(), maNormLoess(x = "maA",
+     y = "maM", z = "maPrintTip")), a.loc = maCompNormEq())
```

We define a function able to compute the Westfall and Young (1993) adjusted p-values for different normalizations of the same Apo batch. For this we use functionality of the `multtest` package (Dudoit et al. (2002)).

```
> library(multtest)
```

```
Loading required package: Biobase
Loading required package: tools

Welcome to Bioconductor

        Vignettes contain introductory material.
        To view, simply type 'openVignette()' or start with 'help(Biobase)'.
        For details on reading vignettes, see the openVignette help page.

Loading required package: survival
Loading required package: splines
```

```
> getP <- function(maObj) {
+     X <- maM(maObj)
+     class <- c(rep(0, 8), rep(1, 8))
+     resT <- mt.maxT(X, class, B = 15000)
+     ord <- order(resT$index)
+     resT$adjp[ord]
+ }
```

Now we get the adjusted p-values for the three situations: raw data (Anone), composite normalization (cpLo2DA) and robust neural networks based (pNN2DA).

```
> pAnone <- log10(getP(Apo))
> pAcPLo2D <- log10(getP(AcPLo2D))
> pApNN2DA <- log10(getP(ApNN2DA))
```

We plot the log of adjusted p-values obtained using the normalized data. Results are shown in Figure 3.

```
> kout <- c(2149, 4139, 5356, 540, 1739, 1496, 2537, 4941)
> methods <- c("none", "cPLo2D", "pNN2DA")
> plot(rep(1, length(pAnone)), pAnone, xlim = c(1, 3), ylim = c(-4,
+     0), xlab = "Method", ylab = "log10(p)", axes = FALSE)
> points(rep(1, 8), pAnone[kout], col = "red", pch = 20)
> points(rep(2, length(pAcPLo2D)), pAcPLo2D)
```

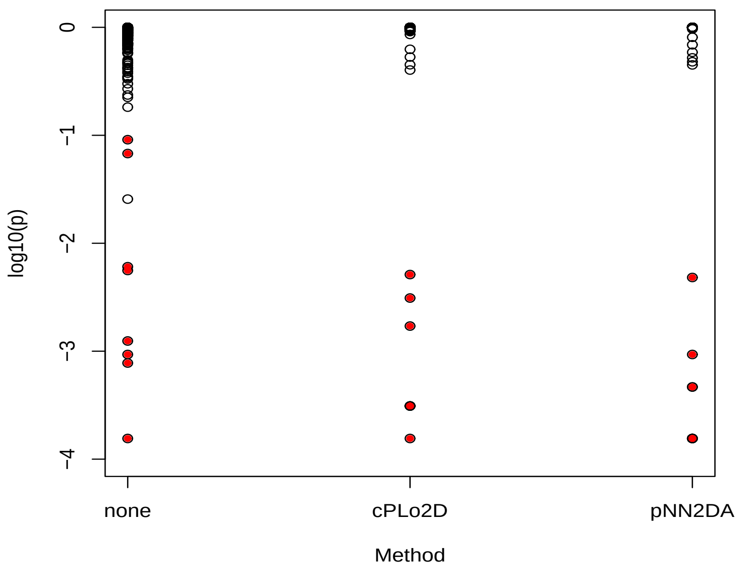


Figure 3: Adjusted p values for raw data (none), composite normalization (cpLo2D), and robust neural networks (pNN2DA)

```
> points(rep(2, 8), pAcPLo2D[kout], col = "red", pch = 20)
> points(rep(3, length(pApNN2DA)), pApNN2DA)
> points(rep(3, 8), pApNN2DA[kout], col = "red", pch = 20)
> axis(1, 1:3, methods)
> axis(2)
> box()
```

The spots with the indices `kout` above are those with smallest t-statistics after loess print tip normalization as in Yang et al. (2001).

Observe in Figure 3 that the 8 genes (represented by filled red circles) may be confidently distinguished from the bulk of non-differentially expressed genes at a threshold of p=0.01. As some of the genes may have received about the same adjusted p-value, less that 8 filled circles may be observed for a particular method.

Now we may want to compare the ability of this new normalization method to reduce the variability of log ratios within slides. A easy way to do that is to look at the distributions of the normalized `M` values. The method `compNorm` of `nnNorm` package will do this (see Figure 4).

```
> compNorm(maM(Apo), maM(AcPLo2D), maM(ApNN2DA), bw = 0.9, xlim = c(-2,
+     2), titles = methods, type = "d")
```

The same distributions may be inspected using the box plot in Figure 5.

```
> compNorm(maM(Apo), maM(AcPLo2D), maM(ApNN2DA), bw = 0.9, xlim = c(-2,
+     2), titles = methods, type = "b")
```

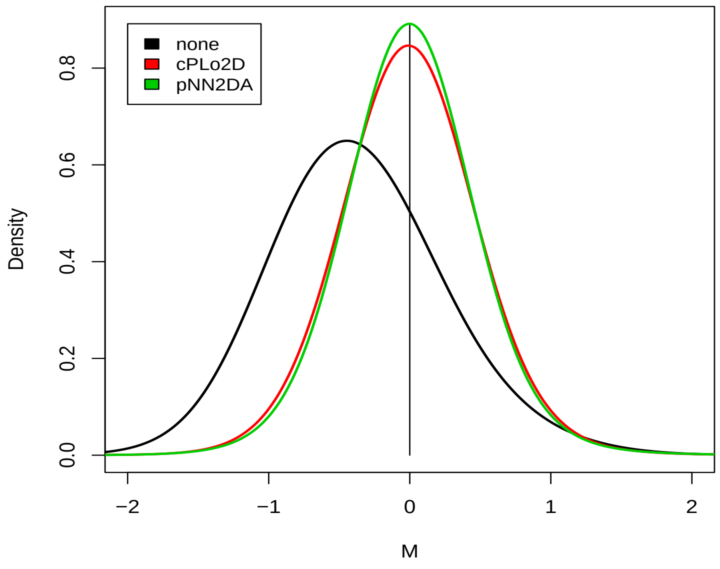


Figure 4: Density plots of normalized log ratios. The highest density curve corresponds to the robust neural networks normalization.

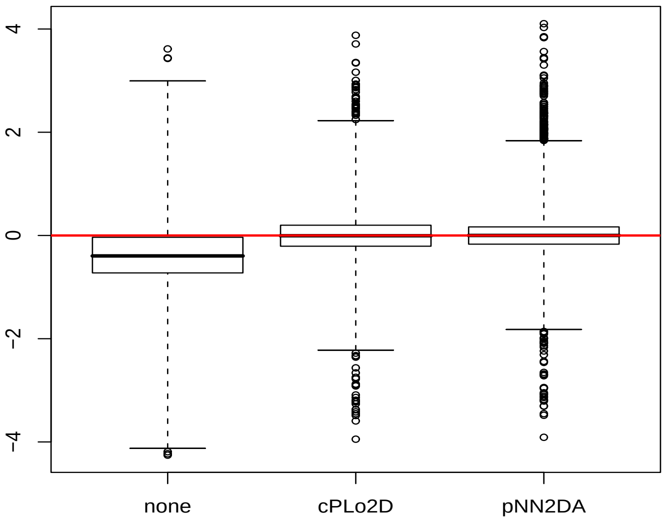


Figure 5: Box plot of normalized log ratios

The maNormNN function in nnNorm package offers flexibility in choosing the type of dependence of the bias to be accounted for. For e.g. setting binHeight parameter as equal with the number of spot rows in a print tip group, will discard the variable Y in the bias model.

```
> ApoNNy <- maNormNN(Apo[, 1], binWidth = 3, binHeight = maNsr(Apo))
```

In this case only A and X will be used as regressors for the bias. Similarly by setting binWidth=maNsc(Apo) in the same function will suppress the contribution of the X variable form the bias estimation. The M-A plots may be shown before and after the normalization for each slide if the parameter maplots is set to "TRUE".

For the case when only intensity normalization is desired (X, and Y regressors discarded) we provide the facility to plot the bias estimates against the intensity level. For e.g. lets perform intensity normalization only for the 13th slide of the Apo batch:

```
> par(mfrow = c(2, 1))
> par(mar = c(2, 4, 2, 2))
> ApoNr <- maNormNN(Apo[, 13], binWidth = maNsc(Apo), binHeight = maNsr(Apo),
+     robust = TRUE, maplots = TRUE)
```

References

M. J. Callow, S. Dudoit, E. L. Gong, T. P. Speed, and E. M. Rubin. Microarray expression profiling identifies genes with altered expression in hdl-deficient mice. *Genome Research*, 10:2022–2029, 2000.

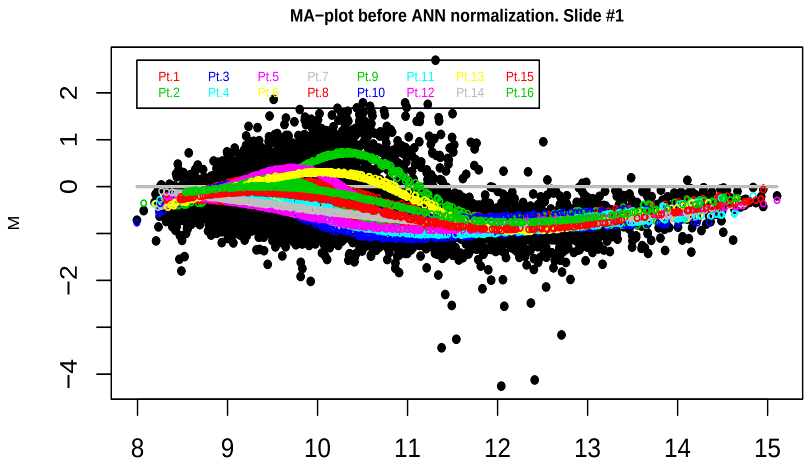
S. Dudoit, J. P. Shaffer, and J. C. Boldrick. Multiple hypothesis testing in microarray experiments. *Statistical Science*, to appear, preprint available at UC Berkeley, Division Biostatistics working paper series: 2002-110, http://www.bepress.com/ucbbiostat/paper110, 2002.

A. L. Tarca, J. E. K. Cooke, and J. Mackay. A robust neural networks approach for spatial and intensity-dependent normalization of cdna microarray data. *Bioinformatics*, 21(11):2674–2683, 2005.

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

Y. Yang, S. Dudoit, P. Luu, and T. Speed. Normalization for cdna microarray data. *SPIE BiOS*, 2001.

Y. Yang, S. Dudoit, P. Luu, D. Lin, V. Peng, J. Ngai, and T. Speed. Normalization for cdna microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res.*, 30:e15, 2002.



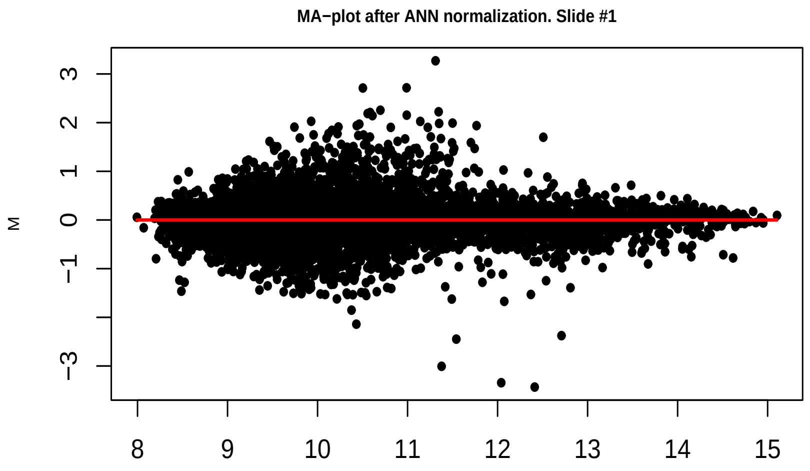


Figure 6: M-A plots before and after normalization for the whole slide.