Normalization: Bioconductor's marray package

Yee Hwa Yang[1] and Sandrine Dudoit[2]

October 5, 2004

1. Department of Medicine, University of California, San Francisco, jean@biostat.berkeley.edu
2. Division of Biostatistics, University of California, Berkeley,
http://www.stat.berkeley.edu/~sandrine

Contents

1 Overview

This document provides a tutorial for the normalization component of the marray package. Greater details on the packages are given in [Dudoit and Yang(2002)]. Like most Bioconductor packages, these four packages rely on the object–oriented class/method mechanism, provided by the R methods package, to allow efficient and systematic representation and manipulation of microarray data.

The package implements robust adaptive location and scale normalization procedures, which correct for different types of dye biases (e.g. intensity, spatial, plate biases) and allow the use of control sequences spotted onto the array and possibly spiked into the mRNA samples. Normalization is needed to ensure that observed differences in intensities are indeed due to differential expression and not experimental artifacts; fluorescence intensities should therefore be normalized before any analysis which involves comparisons among genes within or between arrays.

2 Getting started

Installing the package. To install the *marray* package for Windows operating systems, first start R and make sure you are connected to the internet. Next, select "**Packages**" from the menu and click on " **Install package(s) from Bioconductor...**". Lastly, select *marray* from the pop–up windows and click click on "**OK**". You will find more detailed installation instructions on the Bioconductor web site.
Loading the package. To load the `marray` package in your R session, type `library(marray)`.
Help files. As with any R package, detailed information on functions, classes and methods can be obtained in the help files. For instance, to view the help file for the function `read.GenePix` in a browser, use `help.start()` followed by `? maNorm`.

Case study. We demonstrate the functionality of this collection of R packages using gene expression data from the Swirl zebrafish experiment. These data are included as part of the `marray` package, hence you will also need to install this package. To load the swirl dataset, use `data(swirl)`, and to view a description of the experiments and data, type `? swirl`.

Next. After the two main pre–processing tasks, image analysis and normalization, the next steps in the statistical analysis depend on the biological question for which the microarray experiment was designed. Thus, different Bioconductor packages may be applicable. For example, for identifying differentially expressed genes, functions in the packages `genefilter`, `edd`, `sma`, and `multtest` may be used.

3 Normalization using robust local regression

The purpose of normalization is to identify and remove sources of systematic variation, other than differential expression, in the measured fluorescence intensities (e.g. different labeling efficiencies and scanning properties of the Cy3 and Cy5 dyes; different scanning parameters, such as PMT settings; print–tip, spatial, or plate effects). It is necessary to normalize the fluorescence intensities before any analysis which involves comparing expression levels within or between slides (e.g. classification, multiple testing), in order to ensure that differences in intensities are indeed due to differential expression and not experimental artifacts. The need for normalization can be seen most clearly in self–self experiments, in which two identical mRNA samples are labeled with different dyes and hybridized to the same slide [Dudoit et al.(2002)Dudoit, Yang, Callow, and Speed]. Although there is no differential expression and one expects the red and green intensities to be equal, the red intensities often tend to be lower than the green intensities. Furthermore, the imbalance in the red and green intensities is usually not constant across the spots within and between arrays, and can vary according to overall spot intensity, location on the array, plate origin, and possibly other variables.

Location normalization. We have developed location normalization methods which correct for intensity, spatial, and other dye biases using *robust locally weighted regression* [Cleveland(1979), Yang et al.(2001)Yang, Dudoit, Luu, and Speed, Yang et al.(2002)Yang, Dudoit, Luu, Lin, Peng, Ngai, and Speed
Local regression is a *smoothing* method for summarizing multivariate data using general curves and

surfaces. The smoothing is achieved by fitting a linear or quadratic function of the predictor variables *locally* to the data, in a fashion that is analogous to computing a moving average. In the lowess and loess procedures, polynomials are fitted locally using iterated weighted least squares. *Robust* fitting guards against deviant points distorting the smoothed points. In the context of microarray experiments, robust local regression allows us to capture the non–linear dependence of the intensity log–ratio $M = \log_2 R/G$ on the overall intensity $A = \log_2 \sqrt{RG}$, while ensuring that the computed normalization values are not driven by a small number of differentially expressed genes with extreme log–ratios. For details on the R `loess` function (`modreg` package), type `? loess`.

Scale normalization. For scale normalization, a robust estimate of scale, such as the *median absolute deviation (MAD)*, may be used [Yang et al.(2001)Yang, Dudoit, Luu, and Speed, Yang et al.(2002)Yang, Dudo
For a collection of numbers $x_1, \ldots, x_n$, the MAD is the median of their absolute deviations from the median $m = \text{median}\{x_1, \ldots, x_n\}$

$$MAD = \text{median}\{|x_1 - m|, \ldots, |x_n - m|\}.$$

The R function for MAD is `mad`.

Location and scale normalized intensity log–ratios M are given by

$$M \leftarrow \frac{M - l}{s},$$

where l and s denote the location and scale normalization values, respectively. The location value l can be obtained, for example, by robust local regression of M on A within print–tip–group. The scale value s could be the MAD, within print–tip–group, of location normalized log–ratios.

4 Normalization functions

4.1 General normalization function `maNormMain`

The main function for location and scale normalization of cDNA microarray data is `maNormMain`; it has eight arguments (see also `? maNormMain`):

`mbatch`: Object of class `marrayRaw`, containing intensity data for the batch of arrays to be normalized. An object of class `marrayNorm` may also be passed if normalization is performed in several steps.

`f.loc`: A list of location normalization functions, e.g., `maNormLoess`, `maNormMed`, or `maNorm2D`.

`f.scale`: A list of scale normalization functions, e.g, `maNormMAD`.

`a.loc`: For composite normalization, a function for computing the weights used in combining several location normalization functions, e.g., `maCompNormA`.

`a.scale`: For composite normalization, a function for computing the weights used in combining several scale normalization functions.

`Mloc`: If `TRUE`, the location normalization values are stored in the slot `maMloc` of the object of class `marray` returned by the function, if `FALSE`, these values are not retained. This option allows to save memory for large datasets.

`Mscale`: If `TRUE`, the scale normalization values are stored in the slot `maMscale` of the object of class `marray` returned by the function, if `FALSE`, these values are not retained.

`echo`: If `TRUE`, the index of the array currently being normalized is printed.

Normalization is performed simultaneously for each array in the batch using the location and scale normalization procedures specified by the lists of functions `f.loc` and `f.scale`. Typically, only one function is given in each list, otherwise composite normalization is performed using the weights given by `a.loc` and `a.scale` [Yang et al.(2002)Yang, Dudoit, Luu, Lin, Peng, Ngai, and Speed]. The `maNormMain` function returns objects of class `marray` (for more details on microarray classes, consult the help files and vignettes for the package `marrayClasses`, for example type `? marrrayNorm`).

The `marray` package contains functions for median (`maNormMed`), intensity or A–dependent (`maNormLoess`), and 2D spatial (`maNorm2D`) location normalization. The R robust local regression function `loess` is used for intensity dependent and 2D spatial normalization. The package also contains a function for scale normalization using the median absolute deviation (MAD) (`maNormMAD`). These functions have arguments for specifying which spots to use in the normalization and for controlling the local regression, when applicable. The functions allow normalization to be done separately within values of a layout parameter, such as plate or print–tip–group, and using different subsets of probe sequences (e.g. dilution series of control probe sequences).

4.2 Simple normalization function `maNorm`

A simple wrapper function `maNorm` is provided for users interested in applying a standard set of normalization procedures using default parameters. This function returns an object of class `marray` and has seven arguments

`mbatch`: Object of class `marrayRaw`, containing intensity data for the batch of arrays to be normalized. An object of class `marray` may also be passed if normalization is performed in several steps.

`norm`: Character string specifying the normalization procedure. Six normalization procedures are available with this function: `none`, for no normalization; `median`, for global median location normalization; `loess` for global intensity or A–dependent location normalization using the `loess` function; `twoD`, for 2D spatial location normalization using the `loess` function; `printTipLoess`, for within–print–tip–group intensity dependent location normalization using the `loess` function; and `scalePrintTipMAD`, for within–print–tip–group intensity dependent location normalization followed by within–print–tip–group scale normalization using the median absolute deviation. This argument can be specified using the first letter of each method.

`subset`: A logical or numeric vector indicating the subset of points used to compute the normalization values.

`span`: The argument span which controls the degree of smoothing in the loess function. Only used for `loess`, `twoD`, `printTipLoess`, and `scalePrintTipMAD` options.

`Mloc`: If `TRUE`, the location normalization values are stored in the slot `maMloc` of the object of class `marray` returned by the function, if `FALSE`, these values are not retained. This option allows to save memory for large datasets.

`Mscale`: If `TRUE`, the scale normalization values are stored in the slot `maMscale` of the object of class `marray` returned by the function, if `FALSE`, these values are not retained.

`echo`: If `TRUE`, the index of the array currently being normalized is printed.

4.3 Simple scale normalization function `maNormScale`

A simple wrapper function `maNormScale` is provided for users interested in applying a standard set of scale normalization procedures using default parameters. This function returns an object of class `marrayNorm` has six arguments

`mbatch`: Object of class `marrayRaw`, containing intensity data for the batch of arrays to be normalized. An object of class `marray` may also be passed if normalization is performed in several steps.

`norm`: Character string specifying the normalization procedure. Two normalization procedures are currently available for this function: `globalMAD` for global scale normalization using the median absolute deviation; `printTipMAD` for within–print–tip–group scale normalization using the median absolute deviation. This argument can be specified using the first letter of each method.

`subset`: A logical or numeric vector indicating the subset of points used to compute the normalization values.

`geo`: If `TRUE`, the MAD of each group is divided by the geometric mean of the MADs across groups [Yang et al.(2002)Yang, Dudoit, Luu, Lin, Peng, Ngai, and Speed]. This allows observations to retain their original units.

`Mscale`: If `TRUE`, the scale normalization values are stored in the slot `maMscale` of the object of class `marray` returned by the function, if `FALSE`, these values are not retained.

`echo`: If `TRUE`, the index of the array currently being normalized is printed.

The `globalMad` option, with `geo=TRUE`, allows between slide scale normalization.

5 Normalization of Swirl zebrafish microarray data

To read in the data for the Swirl experiment and generate the plate IDs (see `marrayClasses` and `marrayInput` for greater details)

```
> library("marray", verbose = FALSE)
> data(swirl)
> maPlate(swirl) <- maCompPlate(swirl, n = 384)
```

The pre–normalization MA–plot for the Swirl 93 array in Figure 3 illustrates the non–linear dependence of the log–ratio M on the overall spot intensity A and the existence of spatial dye biases. Only a small proportion of the spots are expected to vary in intensity between the two channels. We thus perform within–print–tip–group loess location normalization using all $8,448$ probes on the array.

5.1 Using main function `maNormMain`

The following command normalizes all four arrays in the Swirl experiment simultaneously

```
> swirl.norm <- maNormMain(swirl, f.loc = list(maNormLoess(x = "maA",
+     y = "maM", z = "maPrintTip", w = NULL, subset = TRUE, span = 0.4)),
+     f.scale = NULL, a.loc = maCompNormEq(), a.scale = maCompNormEq(),
+     Mloc = TRUE, Mscale = TRUE, echo = FALSE)
> swirl.norm

An object of class "marrayNorm"
@maA
     swirl.1.spot swirl.2.spot swirl.3.spot swirl.4.spot
[1,]     14.32811     14.09380    11.412575    14.024738
[2,]     14.57599     14.21926    11.449228    14.460780
[3,]     14.42166     14.18555    11.282760    13.816640
[4,]     13.09658     12.69088     8.148732     9.172988
[5,]     13.06419     12.59373     9.594076     8.285839
8443 more rows ...

@maM
     swirl.1.spot swirl.2.spot swirl.3.spot swirl.4.spot
[1,]    0.2982836  -0.08668386    0.9560849  -0.24356041
[2,]    0.3158471  -0.14244913    0.9662092  -0.09231218
[3,]    0.3863162  -0.06367806    1.0483999  -0.05768779
[4,]    0.5491054   0.20727228    0.3622845  -0.27844191
[5,]    0.5430306   0.06018507    0.5656549  -0.35905889
8443 more rows ...

@maMloc
           [,1]       [,2]       [,3]       [,4]
[1,] -0.4722579 -0.1688564 -0.8652834 -0.2582892
[2,] -0.4337398 -0.1701848 -0.8663450 -0.2445663
[3,] -0.4581902 -0.1698466 -0.8599113 -0.2659071
[4,] -0.5547290 -0.1737234 -0.6378262 -0.2776110
[5,] -0.5526962 -0.1758577 -0.7714374 -0.2169173
8443 more rows ...

@maMscale
```

```
<0 x 0 matrix>

@maW
<0 x 0 matrix>

@maLayout
An object of class "marrayLayout"
@maNgr
[1] 4

@maNgc
[1] 4

@maNsr
[1] 22

@maNsc
[1] 24

@maNspots
[1] 8448

@maSub
[1] TRUE

@maPlate
[1] 1 1 1 1 1
Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
8443 more elements ...

@maControls
[1] Control Control Control Control Control
Levels: Control probes
8443 more elements ...

@maNotes
[1] "No Input File"


@maGnames
An object of class "marrayInfo"
@maLabels
[1] "geno1" "geno2" "geno3" "3XSSC" "3XSSC"
8443 more elements ...
```

```
@maInfo
     "ID" "Name"
1 control  geno1
2 control  geno2
3 control  geno3
4 control  3XSSC
5 control  3XSSC
8443 more rows ...

@maNotes
[1] "C:/GNU/R/rw1041/library/marrayInput/data/fish.gal"


@maTargets
An object of class "marrayInfo"
@maLabels
[1] "81" "82" "93" "94"

@maInfo
  # of slide        Names experiment Cy3 experiment Cy5      date comments
1         81 swirl.1.spot          swirl      wild type 2001/9/20       NA
2         82 swirl.2.spot      wild type          swirl 2001/9/20       NA
3         93 swirl.3.spot          swirl      wild type 2001/11/8       NA
4         94 swirl.4.spot      wild type          swirl 2001/11/8       NA

@maNotes
[1] "C:/GNU/R/rw1041/library/marrayInput/data/SwirlSample.txt"


@maNotes
[1] ""

@maNormCall
maNormMain(mbatch = swirl, f.loc = list(maNormLoess(x = "maA",
    y = "maM", z = "maPrintTip", w = NULL, subset = TRUE, span = 0.4)),
    f.scale = NULL, a.loc = maCompNormEq(), a.scale = maCompNormEq(),
    Mloc = TRUE, Mscale = TRUE, echo = FALSE)
```

This is the default normalization procedure in `maNormMain`, thus the same results could be obtained by calling

```
> swirl.norm <- maNormMain(swirl)
```

To see the effect of within–print–tip–group location normalization, compare the pre–and post–normalization boxplots and MA–plots in Figures 1, 2, and 3. Normalized log–ratios M are now evenly distributed about about zero across the range of intensities A for each print–tip–group.

Furthermore, the non–linear location normalization seems to have eliminated, to some extent, the scale differences among print–tip–groups and arrays.

5.2 Using simple function `maNorm`

Alternately, the simple wrapper function could be used to perform the same normalization

```
> swirl.norm <- maNorm(swirl, norm = "p")
```

For global median normalization

```
> swirl.normm <- maNorm(swirl, norm = "median")
```

5.3 Using simple function `maNormScale`

This simple wrapper function may be used to perform scale normalization separately from location normalization. The following examples do not represent a recommended analysis but are simply used for demonstrating the software functionality. Within–print–tip–group intensity dependent normalization followed by within–print–tip–group scale normalization using the median absolute deviation, could be performed in one step by

```
> swirl.norms <- maNorm(swirl, norm = "s")
```

or sequentially by

```
> swirl.norm1 <- maNorm(swirl, norm = "p")
> swirl.norm2 <- maNormScale(swirl.norm1, norm = "p")
```

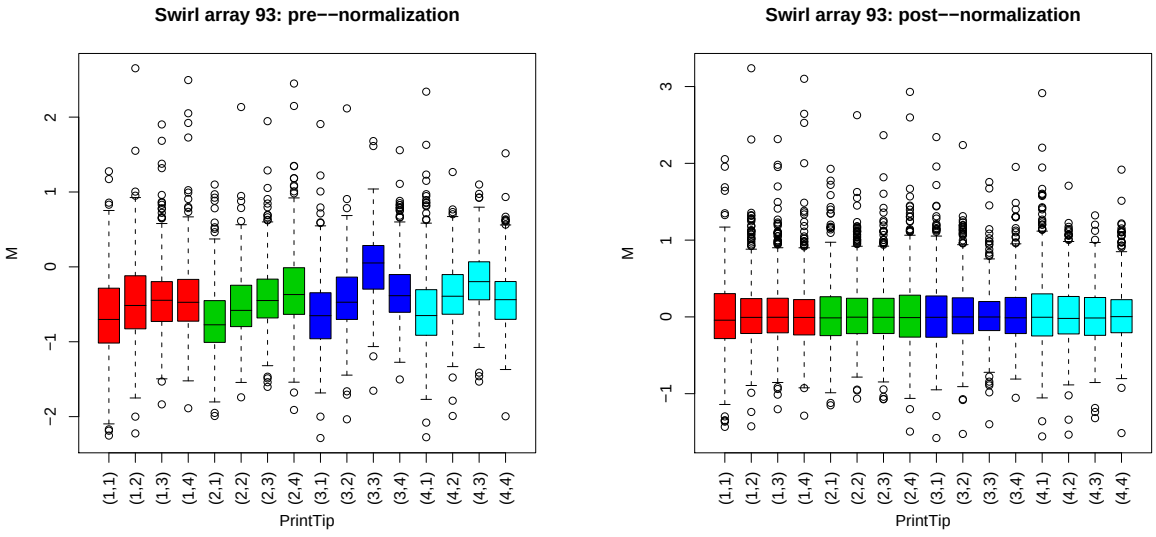
For between slide scale normalization using MAD scaled by the geometric mean of MAD across slides [Yang et al.(2001)Yang, Dudoit, Luu, and Speed, Yang et al.(2002)Yang, Dudoit, Luu, Lin, Peng, Ngai, and Spee

```
> swirl.normg <- maNormScale(swirl.norm, norm = "g")
```

5.4 Plots

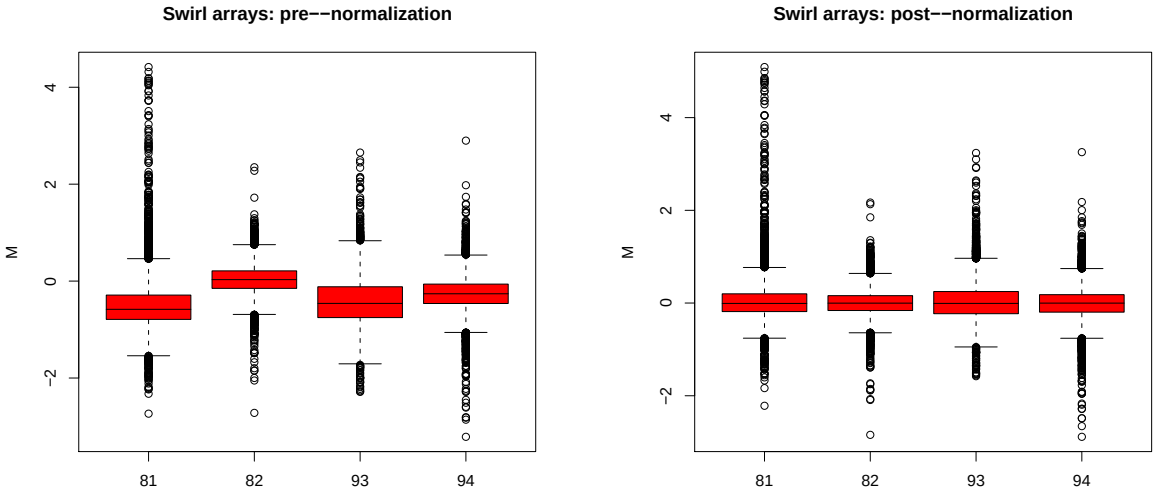
The plots were produced using the following commands, for greater details consult the vignettes `marrayPlots`.

```
> boxplot(swirl[, 3], xvar = "maPrintTip", yvar = "maM", main = "Swirl array 93: pre--normalizat

> boxplot(swirl, yvar = "maM", main = "Swirl arrays: pre--normalization")

> boxplot(swirl.norm[, 3], xvar = "maPrintTip", yvar = "maM", main = "Swirl array 93: post--norm

> boxplot(swirl.norm, yvar = "maM", main = "Swirl arrays: post--normalization")

> plot(swirl[, 3], main = "Swirl array 93: pre--normalization MA--plot")

> plot(swirl.norm[, 3], main = "Swirl array 93: post--normalization MA--plot")
```



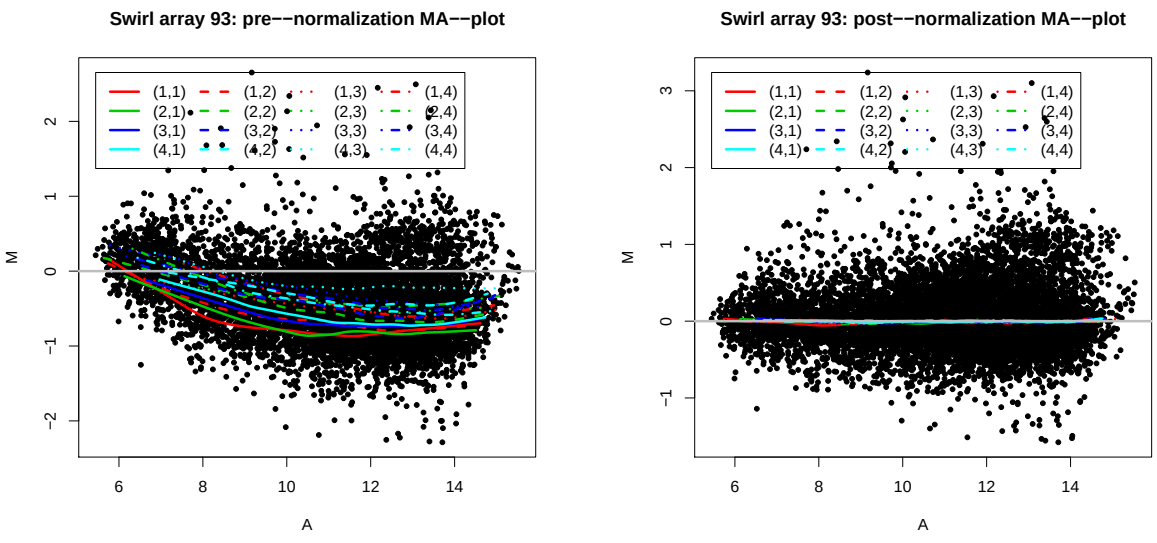
(a) (b)

Figure 1: Boxplots by print–tip–group of the pre– and post–normalization intensity log-ratios M for the Swirl 93 array.



(a) (b)

Figure 2: Boxplots of the pre–and post–normalization intensity log–ratios M for the four arrays in the Swirl experiment.



(a) , (b)

Figure 3: Pre– and post–normalization MA–plot for the Swirl 93 array, with the lowess fits for individual print–tip–groups. Different colors are used to represent lowess curves for print–tips from different rows, and different line types are used to represent lowess curves for print–tips from different columns.

Note: Sweave. This document was generated using the `Sweave` function from the R *tools* package. The source file is in the `/inst/doc` directory of the package *marray*.

References

[Cleveland(1979)] W. S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 74(368):829–836, 1979.

[Dudoit and Yang(2002)] S. Dudoit and Y. H. Yang. Bioconductor R packages for exploratory analysis and normalization of cDNA microarray data. In G. Parmigiani, E. S. Garrett, R. A. Irizarry, and S. L. Zeger, editors, *The Analysis of Gene Expression Data: Methods and Software*. Springer, New York, 2002.

[Dudoit et al.(2002)Dudoit, Yang, Callow, and Speed] S. Dudoit, Y. H. Yang, M. J. Callow, and T. P. Speed. Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statistica Sinica*, 12(1):111–139, 2002.

[Yang et al.(2002)Yang, Dudoit, Luu, Lin, Peng, Ngai, and Speed] Y. H. Yang, S. Dudoit, P. Luu, D. M. Lin, V. Peng, J. Ngai, and T. P. Speed. Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Research*, 30(4), 2002.

[Yang et al.(2001)Yang, Dudoit, Luu, and Speed] Y. H. Yang, S. Dudoit, P. Luu, and T. P. Speed. Normalization for cDNA microarray data. In M. L. Bittner, Y. Chen, A. N. Dorsel, and E. R. Dougherty, editors, *Microarrays: Optical Technologies and Informatics*, volume 4266 of *Proceedings of SPIE*, May 2001.