affyPLM: Model Based QC Assessment of Affymetrix GeneChips

Ben Bolstad
bmb@bmbolstad.com
http://bmbolstad.com

November 11, 2026

Contents

1 Introduction **1**

2 Fitting Probe Level Models for Quality Assessment **2**

3 Quality Diagnostics **2**
3.1 Chip Pseudo-images 2
3.2 RLE 7
3.3 NUSE 8

4 Final Comments **9**

1 Introduction

This document is a basic users guide to the quality assessment facilities of the affyPLM package. Other vignettes for this package describe other functionalities. Quality assessment is something that should be carried out in the initial analysis of any Affymetrix GeneChip dataset. affyPLM provides a number of useful tools based on probe-level modeling procedures.

To begin, load the package using

```
> library(affyPLM)
> options(width=40)
```

2 Fitting Probe Level Models for Quality Assessment

The core function for fitting the probe-level models is `fitPLM`. As input it takes an *AffyBatch* object and outputs a *PLMset* object. Quality assessment functions operate on *PLMset* objects. Note that there is another vignette which describes in much greater deal the `fitPLM`. Note that affyPLM also provides the `rmaPLM` and `threestepPLM` functions which will also take *AffyBatch* objects and return *PLMset* objects. Some quality assessment functions may operate successfully on these *PLMset* objects, but in general for appropriate probe-level model quality assessment use `fitPLM`.

In this vignette we will use the *Dilution* dataset. Note that this is a small test dataset provided by the affydata package. The following code loads this dataset and then fits probe-level models to all of the probesets.

```
> require(affydata)
> data(Dilution)   # an example dataset provided by the affydata package
> #FIXME:remove the next line
> Dilution = updateObject(Dilution)
> Pset <- fitPLM(Dilution)
```

3 Quality Diagnostics

3.1 Chip Pseudo-images

Chip pseudo-images are very useful for detecting artifacts on arrays that could pose potential quality problems. More specifically, weights and residuals from model fitting procedures are stored in *PLMset* objects and may be graphically displayed using the `image` function. When the `image` function is applied to a *PLMset* object a pseudo-image of the chip based upon the residual is produced. By default, chip pseudo-image of the weights are produced. The following code produces a chip pseudo image of the weights for the second array in the dataset.

```
> image(Pset,which=2)
```

Note that the `which` argument is used to control which array is drawn. Its value should be between 1 and n, where n is the number of arrays in the dataset. If `which` is not supplied, then images for all chips are drawn, one by one in succession. Figure 1 shows the resultant chip pseudo image. Areas of low weight are greener, high weights (ie weight of 1) are light grey.

While the conventional coloring scheme for weights pseudo chip images used terrain coloring, it is possible to override this by supplying the `col` argument to `image`. The following code creates weights images using two different greyscale (black to white) and (white to black):

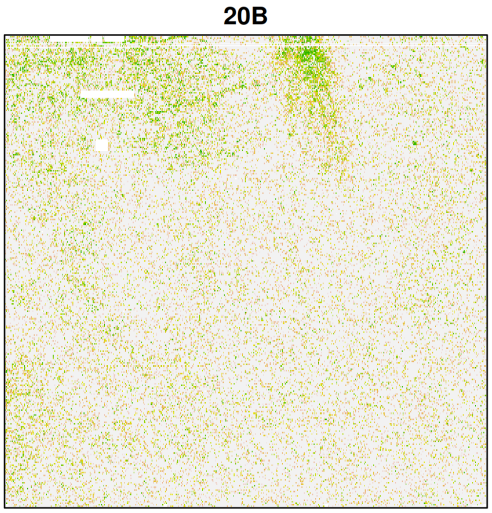


Figure 1: An image of the weights for the second chip in the dataset

```
> image(Pset,which=2,col=gray(0:25/25),add.legend=TRUE)
> image(Pset,which=2,col=gray(25:0/25),add.legend=TRUE)
```

with Figure 2 showing the results. Note that it is recommended that you use a legend if you use an alternative coloring scheme. Legends are added when `add.legend=TRUE` is supplied.

Residuals are the second quantity for which affyPLM produces chip pseudo images. Four different views of residuals are provided: residuals, positive residuals, negative residuals and sign of residuals. The `type` is used to control which of these images is drawn. Setting `type="resids"` gives a chip pseudo image of residuals, with higher red intensities corresponding with higher positive residuals, white corresponding with residuals close to 0 and more intense blues corresponding with high negative residuals. When `type="pos.resids"` only high positive residuals are drawn in red, with negative and near 0 residuals being drawn in white. Using `type="neg.resids"` only extreme negative residuals are drawn in blue, with positive negative and near 0 residuals being drawn in white. Finally, `type="sign.resids"` gives images where all negative residuals regardless of magnitude are indicated by blue and all positive residuals by red. The following code produces these images, as shown in Figure 3, for the second chip in the *Dilution* dataset:

```
> image(Pset,which=2, type="resids")
> image(Pset,which=2, type="pos.resids")
> image(Pset,which=2, type="neg.resids")
> image(Pset,which=2, type="sign.resids")
```

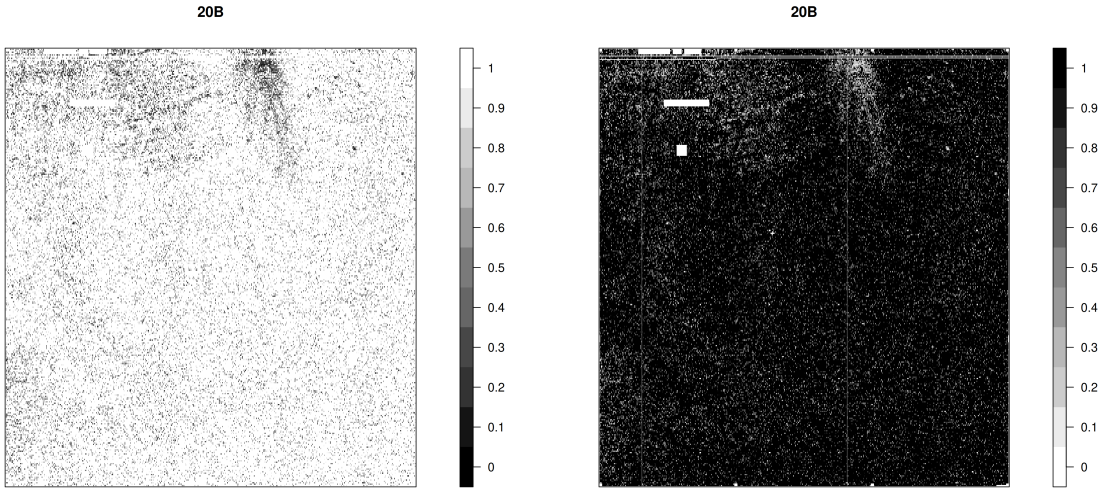


Figure 2: Image of the weights for the second chip in the dataset using different colorings.

One thing that should be noted about the residual chips pseudo images is that a logarithmic color space is used by default. This tends to intensify the coloring of large residuals without highlighting small residuals. Supplying `use.log=FALSE` to the call to `image` will turn this off, but tends to give duller images.

As with chip pseudo images of weights, it is possible to use alternative color schemes for residual images. It is recommended that you use the `pseudoPalette` function to control your color palette in this case. This function creates a color space that moves from one color to another. Some examples are given by the following code, with the results shown in Figure 4:

```
> image(Pset,which=2,type="resids",col=pseudoPalette(low="darkgreen",high="magenta",mid
> image(Pset,which=2,type="pos.resids",col=pseudoPalette(low="yellow",high="darkblue"),
```

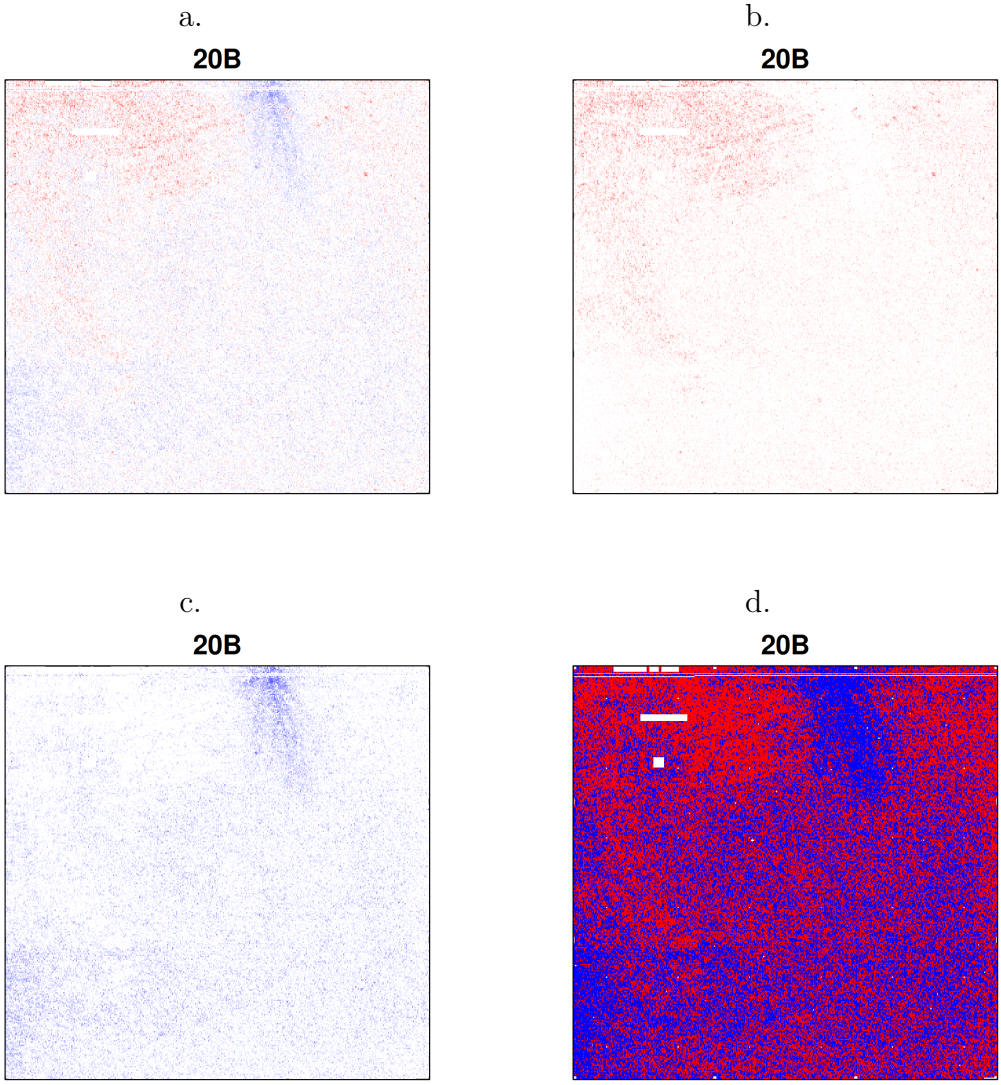


Figure 3: Various chip pseudo-images of residuals. a) Residuals b) Positive residuals c) Negative residuals d) Sign of Residuals

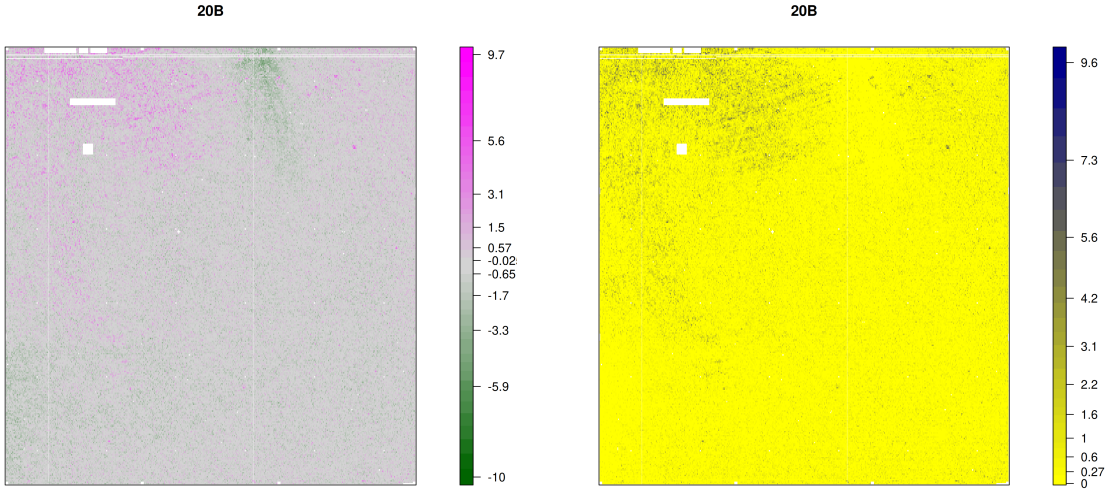


Figure 4: Chip pseudo-images of residuals using different coloring.

3.2 RLE

Another quality assessment tool are Relative Log Expression (RLE) values. Specifically, these RLE values are computed for each probeset by comparing the expression value on each array against the median expression value for that probeset across all arrays. Assuming that most genes are not changing in expression across arrays means ideally most of these RLE values will be near 0. Boxplots of these values, for each array, provides a quality assessment tool. A RLE boxplot can be produced using:

```
> RLE(Pset,main="RLE for Dilution dataset")
```

with the resulting plot shown in Figure 5. When examining this plot focus should be on the shape and position of each of the boxes. Typically arrays with poorer quality show up with boxes that are not centered about 0 and/or are more spread out. For this particular dataset there is no such array.

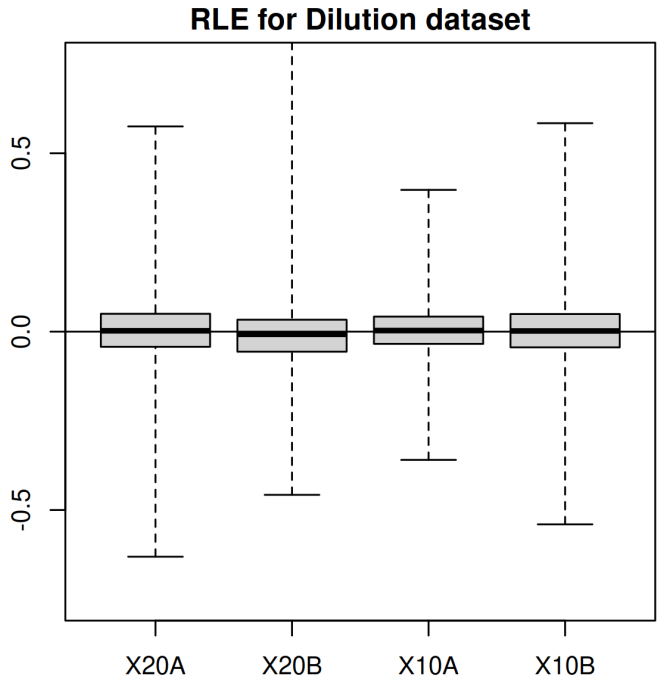


Figure 5: RLE boxplot.

As an alternative to the RLE boxplot it is possible to compute summary statistics, by array, using the following command:

```
> RLE(Pset,type="stats")
```

```
                20A           20B
median 0.002359773 -0.007542337
```

```
IQR    0.092560253  0.089580148
                10A          10B
median 0.003051789 0.001902185
IQR    0.076335304 0.093406833
```

the median and IQR of the RLE values for each array is returned. Note that calling `RLE(Pset,type=values)` returns all the RLE expression values.

3.3 NUSE

Normalized Unscaled Standard Errors (NUSE) can also be used for assessing quality. In this case, the standard error estimates obtained for each gene on each array from `fitPLM` are taken and standardized across arrays so that the median standard error for that genes is 1 across all arrays. This process accounts for differences in variability between genes. An array were there are elevated SE relative to the other arrays is typically of lower quality. Boxplots of these values, separated by array can be used to compare arrays. The `NUSE` function will produce such a plot:

```
> NUSE(Pset,main="NUSE for Dilution dataset")
```

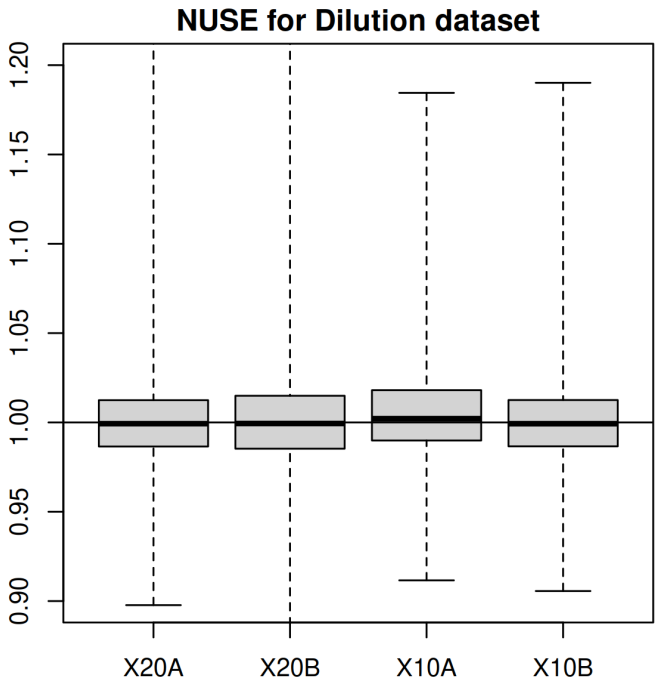


Figure 6: NUSE boxplot.

As an alternative to the NUSE boxplot it is possible to compute summary statistics, by array, using the following command:

```
> NUSE(Pset,type="stats")
```

```
              20A        20B        10A
median 0.99929569 0.99940108 1.00212616
IQR    0.02592197 0.02962219 0.02816366
              10B
median 0.99927854
IQR    0.02591953
```

the median and IQR of the NUSE values for each array is returned. Note that calling `NUSE(Pset,type=values)` returns all the NUSE values.

4 Final Comments

The tools discussed here are most useful for assessing relative quality within a dataset. A typical use of these tools would be to decide whether or not any arrays should be excluded from down-stream data analysis. Some users might find it useful to see the results of these quality assessment procedures on other datasets. Such information can be found on the PLM Image Gallery website `http://plmimagegallery.bmbolstad.com`.