7659 HW2

Guannan Shen

September 20, 2018

Table of Contents

[Quality Control 1](#_Toc525822469)

[a. read in CEL files 2](#_Toc525822470)

[b. Plot the raw microarray images using image() on the object Data. Comment on what you see in these plots. 5](#_Toc525822471)

[c. Plot quality control metrics using qc.affy() and plot.qc.stats() from the simpleaffy package. 5](#_Toc525822472)

[In the following graphs, the color scheme is as following: 7](#_Toc525822473)

[f. Based on the summaries and figures you generated, would you recommend that one or more chips be removed from the analysis? 12](#_Toc525822474)

[Normalization 12](#_Toc525822475)

[In the following graphs, the color scheme is as following: 12](#_Toc525822476)

[(a) Create log transformed data and plot the density before and after log transforming using plotDensity.AffyBatch. Comment on these plots. 12](#_Toc525822477)

[(b) 13](#_Toc525822478)

[Using expresso(), try different normalize.method options 17](#_Toc525822479)

[When using (quantiles, loess, constant) 17](#_Toc525822480)

[summary.method options (avgdiff, mas, medianpolish), keeping the other arguments set 17](#_Toc525822481)

[perfect-match correction method 17](#_Toc525822482)

[d. Get present and absent calls Mas 5.0 using mas5calls() 22](#_Toc525822483)

[(e) If you think there are any problematic chip(s) based on your quality 23](#_Toc525822484)

# Quality Control

1. Install the affy and simpleaffy packages from Bioconductor.
2. Download the CEL files directory from Canvas.
3. These data are from human cell lines treated with low or high levels of a treatment.
4. The meta-data is provided in targets.txt in the celfiles directory.

## a. read in CEL files

Read the 8 CEL files in the directory celfiles using the following functions from the affy package:

library(affy)

## Loading required package: BiocGenerics

## Loading required package: parallel

##   
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':  
##   
## clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,  
## clusterExport, clusterMap, parApply, parCapply, parLapply,  
## parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:stats':  
##   
## IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':  
##   
## anyDuplicated, append, as.data.frame, cbind, colMeans,  
## colnames, colSums, do.call, duplicated, eval, evalq, Filter,  
## Find, get, grep, grepl, intersect, is.unsorted, lapply,  
## lengths, Map, mapply, match, mget, order, paste, pmax,  
## pmax.int, pmin, pmin.int, Position, rank, rbind, Reduce,  
## rowMeans, rownames, rowSums, sapply, setdiff, sort, table,  
## tapply, union, unique, unsplit, which, which.max, which.min

## Loading required package: Biobase

## Welcome to Bioconductor  
##   
## Vignettes contain introductory material; view with  
## 'browseVignettes()'. To cite Bioconductor, see  
## 'citation("Biobase")', and for packages 'citation("pkgname")'.

library(simpleaffy)

## Loading required package: genefilter

## Loading required package: gcrma

# get the working directory  
getwd()

## [1] "C:/Users/hithr/Documents/Stats/CIDA\_OMICs/7659Stats\_Genetics/HW2/celfiles"

# the pheno Data  
pd <- read.AnnotatedDataFrame("targets.txt", header = TRUE, row.names = 1,   
 as.is = TRUE)  
# the affy data, READ CEL files into Affybatch  
affy\_data <- ReadAffy(filenames = pData(pd)$FileName, phenoData = pd,   
 sampleNames = sampleNames(pd))  
  
  
## extracting information  
head(exprs(affy\_data), 6)

## High1 High2 High3 High4 Low1 Low2 Low3 Low4  
## 1 197 81 106 69 174 103 100 100  
## 2 13571 9974 9267 6591 7231 4831 4724 4724  
## 3 248 137 100 94 194 147 137 137  
## 4 13810 9895 9550 6866 7407 4886 4919 4919  
## 5 123 63 60 76 77 79 71 71  
## 6 173 73 61 68 116 81 103 103

## sampleNames are the same  
sampleNames(pd)

## [1] "High1" "High2" "High3" "High4" "Low1" "Low2" "Low3" "Low4"

sampleNames(affy\_data)

## [1] "High1" "High2" "High3" "High4" "Low1" "Low2" "Low3" "Low4"

## probeNames  
head(probeNames(affy\_data), 6)

## Warning: replacing previous import 'AnnotationDbi::tail' by 'utils::tail'  
## when loading 'hgu133plus2cdf'

## Warning: replacing previous import 'AnnotationDbi::head' by 'utils::head'  
## when loading 'hgu133plus2cdf'

##

## [1] "1007\_s\_at" "1007\_s\_at" "1007\_s\_at" "1007\_s\_at" "1007\_s\_at" "1007\_s\_at"

## mm mismatch  
head(mm(affy\_data), 6)

## High1 High2 High3 High4 Low1 Low2 Low3 Low4  
## 370871 338 123 115 113 131 152 145 145  
## 564482 831 435 393 397 695 791 852 852  
## 1050513 1190 520 345 583 685 688 1033 1033  
## 239977 1288 691 592 564 837 846 891 891  
## 1141565 2740 1161 1035 1477 1540 1938 2269 2269  
## 1131946 786 376 335 355 591 631 791 791

## pm perfect match  
head(pm(affy\_data), 6)

## High1 High2 High3 High4 Low1 Low2 Low3 Low4  
## 369707 368 220 216 248 405 379 476 476  
## 563318 1350 687 590 585 1455 1626 1765 1765  
## 1049349 1414 674 599 586 1464 1422 1701 1701  
## 238813 3838 1830 1472 1446 2980 3205 3701 3701  
## 1140401 3018 1338 1165 1445 2208 2616 3151 3151  
## 1130782 1652 783 668 700 1249 1520 1784 1784

#   
pData(affy\_data)

## FileName Target  
## High1 High\_1\_HG-U133\_Plus\_2.CEL High  
## High2 High\_2\_HG-U133\_Plus\_2.CEL High  
## High3 High\_3\_HG-U133\_Plus\_2.CEL High  
## High4 High\_4\_HG-U133\_Plus\_2.CEL High  
## Low1 Low\_1\_HG-U133\_Plus\_2.CEL Low  
## Low2 Low\_2\_HG-U133\_Plus\_2.CEL Low  
## Low3 Low\_3\_HG-U133\_Plus\_2.CEL Low  
## Low4 Low\_4\_HG-U133\_Plus\_2.CEL Low

pData(pd)

## FileName Target  
## High1 High\_1\_HG-U133\_Plus\_2.CEL High  
## High2 High\_2\_HG-U133\_Plus\_2.CEL High  
## High3 High\_3\_HG-U133\_Plus\_2.CEL High  
## High4 High\_4\_HG-U133\_Plus\_2.CEL High  
## Low1 Low\_1\_HG-U133\_Plus\_2.CEL Low  
## Low2 Low\_2\_HG-U133\_Plus\_2.CEL Low  
## Low3 Low\_3\_HG-U133\_Plus\_2.CEL Low  
## Low4 Low\_4\_HG-U133\_Plus\_2.CEL Low

## use message not cat()  
message("exprs() & mm() & pm() \n& probeNames() \nis not for AnnotateDataFrame 'pd' ")

## exprs() & mm() & pm()   
## & probeNames()   
## is not for AnnotateDataFrame 'pd'

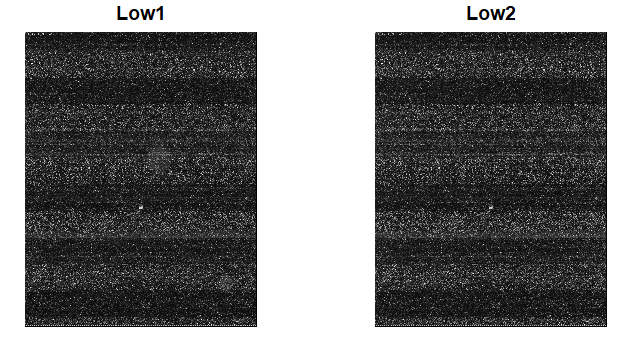
## b. Plot the raw microarray images using image() on the object Data. Comment on what you see in these plots.

This study is to compare the gene expression between two groups via microarray experiment. There are 4 samples in each group, 8 chips in total. From the .CEL file, we can get some ideas of the quality of the raw image .DAT file.  
The first impression is the black background on each chip. On average, it seems like the background of the “High” group is darker than the “Low” group. What’s more, the “High” group has better quality, overall. Because there is neither big dard clot nor obvious aggregated white dot exits and all four chips looks similar. The “Low” group has more within group variability. “Low 1” has several grey clots, although that obvious. “Low 3” seems to have lighter backgroud. The last but not the least, the plot of “Low 4” suggests this chip was defective. This chip has one big dark clot, which makes that area has equal intensity, and the left upper side also exists a vertical black line, which forces that area becomes the black background.  
In summary, these plots suggest there is a problem in the quality of the “Low” group, which requires further quality control diagnostic.

# par(mfrow = c(1,2)) image(affy\_data)

## 

## 

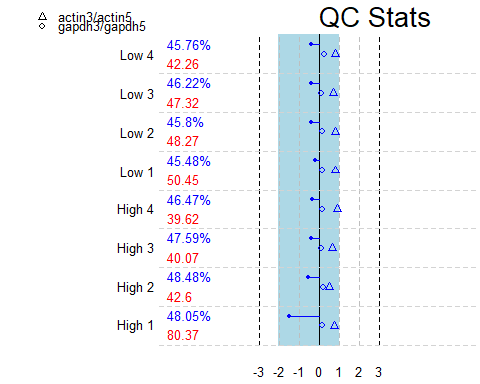


## 

## c. Plot quality control metrics using qc.affy() and plot.qc.stats() from the simpleaffy package.

The QC stats plot reports quality control parameters for the chips. Each array is represented by a seperate row in the figure. In short, everything in the figure should be blue and red highlights a problem.  
On the left hand side of the plot, “%” represents the percentage of genes called present on each array. The blue color means the “present%” are within 10% of each other.It means the percentages of present are close. The red number below each “present%” represents the average background. They are colored red, means there is more than 20 units different among those average background values. Basically, we expect the “present%” and average background to be similar. The central black vertical line corresponding to 0 fold change, the dotted lines on each side represent 3 fold up and down changes. The blue region represents the three-fold change in scale factor of each chip, within these 8 chips, 1.5 fold up and down. The blue bars indicate all chips are in safe scaling region and comparable. The triangle and circle represent beta-actin and GAPDH, house-keeping genes, respectively. Affy state that beta actin should be within 3, gapdh around 1. Blue means these values are in the good region. It is interesting that, the “High 1” sample has the most different overall expression intensity. In summary, less experimental variation among these chips, the better quality they have.

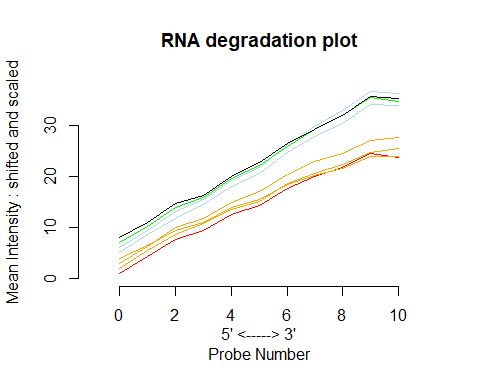
plot.qc.stats(qc.affy(affy\_data), label = c("High 1", "High 2",   
 "High 3", "High 4", "Low 1", "Low 2", "Low 3", "Low 4"))

 ## In the following graphs, the color scheme is as following: “High1”, “High2”, “High3” ,“High4”, “Low1” , “Low2”, “Low3”, “Low4”  
c(“red”,“orange”,“orange”,“orange”,“light blue”,“light blue”,“green”,“black”) ## d. Plot the mean intensity from 3’ to 5’ end of the target mRNA using AffyRNAdeg() and plotAffyRNAdeg(). The RNA digestion plot reports RNA degradation from 5’ to 3’ end of the genes. Usually, the line goes up as moving towards the 3’ end, which indicates more degradation occurs. Ideally, the lines would lie horizontally.  
In practice, we expect the slopes and profiles of the lines are similar for all chips in the study. However, the plot here suggests there is a overall difference between “High” and “Low” groups. The “Low” group has more RNA degradation occured. This indicates there is system error between these two groups happened during the experiment process.

AffyRNAdeg(affy\_data)$sample.names

## [1] "High1" "High2" "High3" "High4" "Low1" "Low2" "Low3" "Low4"

plotAffyRNAdeg(AffyRNAdeg(affy\_data), cols = c("red", "orange",   
 "orange", "orange", "light blue", "light blue", "green",   
 "black"))



# 'High1', 'High2', 'High3' ,'High4', 'Low1' , 'Low2',  
# 'Low3', 'Low4'

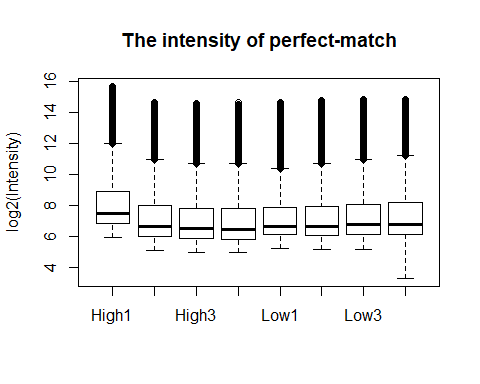
## In the following graphs, the color scheme is as following:

“High1”, “High2”, “High3” ,“High4”, “Low1” , “Low2”, “Low3”, “Low4”  
c(“red”,“orange”,“orange”,“orange”,“light blue”,“light blue”,“green”,“black”) ## e. Use boxplot() and plotDensity.AffyBatch() to examine the distribution of intensity values for the perfect-match and mis-match probes separately. The boxplots and plotDensity just present the same information in two different ways.  
The comparison between the perfect-match (pm) and mis-match (mm) shows:  
1. pm has a higher intensity level on average.  
2. pm has a greater within sample variability, and peaks are wider, larger box and longer tail. 3. The trend of pm intensity across samples is similar to the trend of mm.  
The comparison across samples shows:  
1. The “High 1” (red) has the highest overall intensity. 2. “Low 4” group has clearly more genes with very low intensity, which is abnormal. This is also shown as the long tail at the left handside on the black dotted curve. 3. Besides the “Low 4” group, the spreads (variability) of other samples are similar. 4. Besides the “High 1” group, other medians of intensity are close, around 7 in log2 scale.  
5. The average intensity of “low” group from low 1 to low 4 are slightly higher than high 2 to high 4.

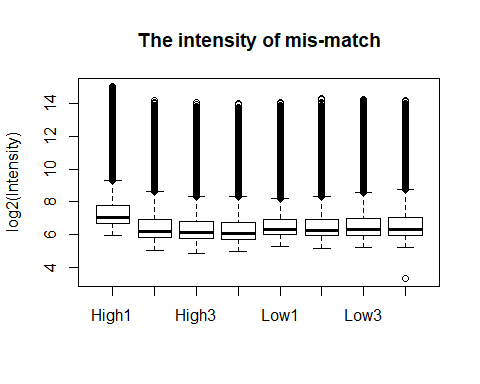
str(affy\_data)

## Formal class 'AffyBatch' [package "affy"] with 10 slots  
## ..@ cdfName : chr "HG-U133\_Plus\_2"  
## ..@ nrow : Named int 1164  
## .. ..- attr(\*, "names")= chr "Rows"  
## ..@ ncol : Named int 1164  
## .. ..- attr(\*, "names")= chr "Cols"  
## ..@ assayData :<environment: 0x0000000038a19a08>   
## ..@ phenoData :Formal class 'AnnotatedDataFrame' [package "Biobase"] with 4 slots  
## .. .. ..@ varMetadata :'data.frame': 2 obs. of 1 variable:  
## .. .. .. ..$ labelDescription: chr [1:2] "" ""  
## .. .. .. ..- attr(\*, "provenance")= chr "Read from file targets.txt on DESKTOP-6711BE0 at Thu Sep 27 14:26:14 2018."  
## .. .. ..@ data :'data.frame': 8 obs. of 2 variables:  
## .. .. .. ..$ FileName: chr [1:8] "High\_1\_HG-U133\_Plus\_2.CEL" "High\_2\_HG-U133\_Plus\_2.CEL" "High\_3\_HG-U133\_Plus\_2.CEL" "High\_4\_HG-U133\_Plus\_2.CEL" ...  
## .. .. .. ..$ Target : chr [1:8] "High" "High" "High" "High" ...  
## .. .. ..@ dimLabels : chr [1:2] "sampleNames" "sampleColumns"  
## .. .. ..@ .\_\_classVersion\_\_:Formal class 'Versions' [package "Biobase"] with 1 slot  
## .. .. .. .. ..@ .Data:List of 1  
## .. .. .. .. .. ..$ : int [1:3] 1 1 0  
## ..@ featureData :Formal class 'AnnotatedDataFrame' [package "Biobase"] with 4 slots  
## .. .. ..@ varMetadata :'data.frame': 0 obs. of 1 variable:  
## .. .. .. ..$ labelDescription: chr(0)   
## .. .. ..@ data :'data.frame': 1354896 obs. of 0 variables  
## .. .. ..@ dimLabels : chr [1:2] "featureNames" "featureColumns"  
## .. .. ..@ .\_\_classVersion\_\_:Formal class 'Versions' [package "Biobase"] with 1 slot  
## .. .. .. .. ..@ .Data:List of 1  
## .. .. .. .. .. ..$ : int [1:3] 1 1 0  
## ..@ experimentData :Formal class 'MIAME' [package "Biobase"] with 13 slots  
## .. .. ..@ name : chr ""  
## .. .. ..@ lab : chr ""  
## .. .. ..@ contact : chr ""  
## .. .. ..@ title : chr ""  
## .. .. ..@ abstract : chr ""  
## .. .. ..@ url : chr ""  
## .. .. ..@ pubMedIds : chr ""  
## .. .. ..@ samples : list()  
## .. .. ..@ hybridizations : list()  
## .. .. ..@ normControls : list()  
## .. .. ..@ preprocessing :List of 2  
## .. .. .. ..$ filenames : chr [1:8] "High\_1\_HG-U133\_Plus\_2.CEL" "High\_2\_HG-U133\_Plus\_2.CEL" "High\_3\_HG-U133\_Plus\_2.CEL" "High\_4\_HG-U133\_Plus\_2.CEL" ...  
## .. .. .. ..$ affyversion: chr NA  
## .. .. ..@ other :List of 1  
## .. .. .. ..$ : chr ""  
## .. .. ..@ .\_\_classVersion\_\_:Formal class 'Versions' [package "Biobase"] with 1 slot  
## .. .. .. .. ..@ .Data:List of 2  
## .. .. .. .. .. ..$ : int [1:3] 1 0 0  
## .. .. .. .. .. ..$ : int [1:3] 1 1 0  
## ..@ annotation : chr "hgu133plus2"  
## ..@ protocolData :Formal class 'AnnotatedDataFrame' [package "Biobase"] with 4 slots  
## .. .. ..@ varMetadata :'data.frame': 1 obs. of 1 variable:  
## .. .. .. ..$ labelDescription: chr NA  
## .. .. ..@ data :'data.frame': 8 obs. of 1 variable:  
## .. .. .. ..$ ScanDate: chr [1:8] "03/14/07 13:29:30" "03/14/07 13:29:30" "03/14/07 13:41:14" "03/14/07 14:27:36" ...  
## .. .. ..@ dimLabels : chr [1:2] "sampleNames" "sampleColumns"  
## .. .. ..@ .\_\_classVersion\_\_:Formal class 'Versions' [package "Biobase"] with 1 slot  
## .. .. .. .. ..@ .Data:List of 1  
## .. .. .. .. .. ..$ : int [1:3] 1 1 0  
## ..@ .\_\_classVersion\_\_:Formal class 'Versions' [package "Biobase"] with 1 slot  
## .. .. ..@ .Data:List of 4  
## .. .. .. ..$ : int [1:3] 3 4 2  
## .. .. .. ..$ : int [1:3] 2 38 0  
## .. .. .. ..$ : int [1:3] 1 3 0  
## .. .. .. ..$ : int [1:3] 1 2 0

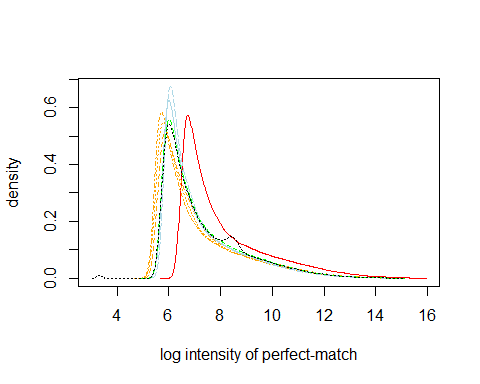
boxplot(log2(pm(affy\_data)), main = "The intensity of perfect-match",   
 ylab = "log2(Intensity)")



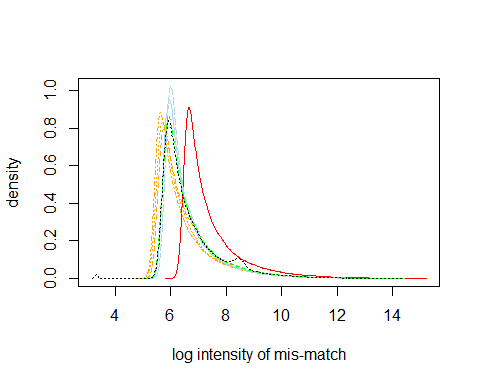
boxplot(log2(mm(affy\_data)), main = "The intensity of mis-match",   
 ylab = "log2(Intensity)")



plotDensity.AffyBatch(affy\_data, which = "pm", xlab = "log intensity of perfect-match",   
 col = c("red", "orange", "orange", "orange", "light blue",   
 "light blue", "green", "black"))



plotDensity.AffyBatch(affy\_data, which = "mm", xlab = "log intensity of mis-match",   
 col = c("red", "orange", "orange", "orange", "light blue",   
 "light blue", "green", "black"))



## f. Based on the summaries and figures you generated, would you recommend that one or more chips be removed from the analysis?

I would recommend to remove the “Low 4” group. This chip is clearly defective (.CEL picture) and result in abnormal intensity values (Intensity plots). Thus, comparing this chip to others is biased.

# Normalization

## In the following graphs, the color scheme is as following:

“High1”, “High2”, “High3” ,“High4”, “Low1” , “Low2”, “Low3”, “Low4”  
c(“red”,“orange”,“orange”,“orange”,“light blue”,“light blue”,“green”,“black”)

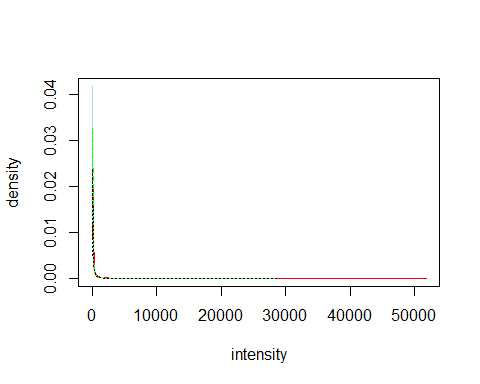
## (a) Create log transformed data and plot the density before and after log transforming using plotDensity.AffyBatch. Comment on these plots.

Without log transformation, the density plots are not readable. Because, majority of genes are at relatively low intensity level, but there are still a decent amount of genes have extreme high intensity.  
After the log2 transformation, the lines are distinguishable. The average intensities of “low” group from low 1 to low 4 are slightly higher than high 2 to high 4. Among those 8 lines, the red line, “High 1” has clearly much higher average intensity than others. The peaks of low group are relatively sharper. The concave shape in the black dotted line (low 4) and the long tail at the left handside indicate some error in this chip.

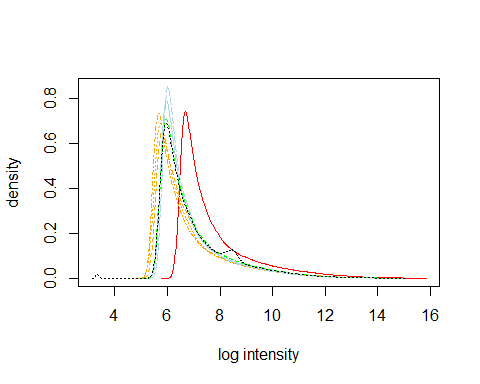
## (b)

From the median/IQR and the shape of the MA plots, the first thing should be noticed that “Low 4” sample has abnormal pattern, indicating artificial error, which is a big problem. The variance (IQR) of “high” group is greater than “Low” group, which means higher variability in the former group. The median intensity in samples high 2 to high 4 are consistently lower than the median of the “Low” group. The shapes are different between these two groups. Generally, the “high” group are more spread out vertically when the average intensity is lower.

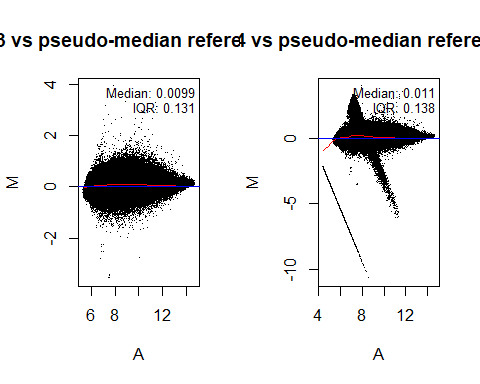
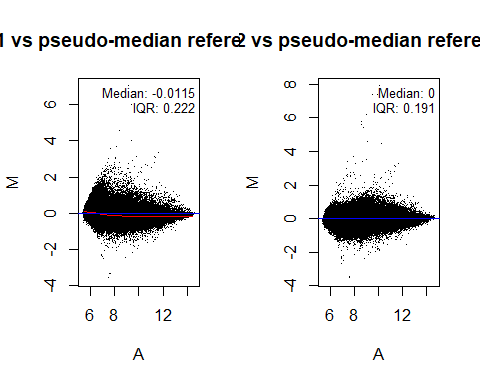
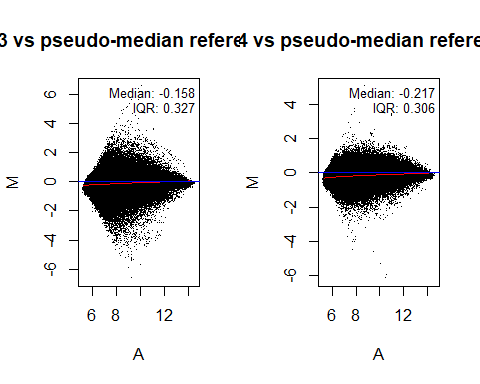
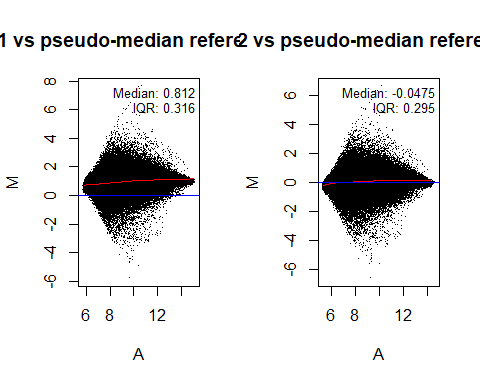
plotDensity.AffyBatch(affy\_data, which = "both", log = FALSE,   
 col = c("red", "orange", "orange", "orange", "light blue",   
 "light blue", "green", "black"))



plotDensity.AffyBatch(affy\_data, which = "both", col = c("red",   
 "orange", "orange", "orange", "light blue", "light blue",   
 "green", "black"))



par(mfrow = c(1, 2))  
MAplot(affy\_data, cex = 0.8)



## Using expresso(), try different normalize.method options

### When using (quantiles, loess, constant)

As we talked in the journal club 2, the quantiles normalization method should be the most aggressive method. In my case, I chose background correction method, RMA, which is widely used in Affymetrix chip. RMA convolution method is to correct the background through the empirical distribution of probe intensities. Summary method is average difference summarization, which is the most commonly used. On each probe set, this method take the average difference between the perfect-match and mis-match. I just use PM only (no correction).  
Compare with un-normalized and other methods, the quantiles normalization gives the most clean normalization, which means all medians and all variance within each sample are quite close. In terms of similar medians and variance, the loess method is the second best method. Meanwhile, the constant normalization can make all medians similar, but the difference in the variance is more obvious, compared with other methods.

### summary.method options (avgdiff, mas, medianpolish), keeping the other arguments set

Summarization is the process that combine the multiple probe intensities for each probe set to generate an expression measurement.  
I used the best normalization method, quantiles for later comparison; background correction method, RMA; PM only (no correction). Three different summary methods are average difference, median polish (RMA) and MAS 5.0. Median polish is using RMA convolution linear model. MAS 5.0 method is using ((PM-MM)/(PM+MM)). MAS option gives a similar result as the average difference method, since both methods share similar underlying math equation. The median polish fits linear model in log2 scale, thus the scale of y-axis are different in this option. Besides the scale, the distribution of the intensity under this option is much more symmetric compared with other two summary methods.

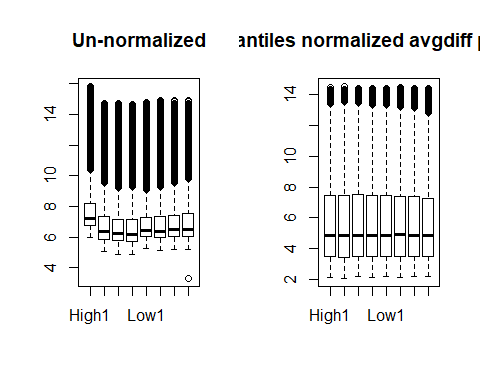
### perfect-match correction method

Quantiles method, RMA background correction, summary method is average difference.  
Subtract mm is just to subtract mis-match from perfect-match to do the correction. The idea of MAS 5.0 is mentioned above. Since the set of RMA method is the best. So I used RMA for background correction.  
The subtractmm and MAS methods are compatible only with the MAS background correction method; that is, it does not make sense to combine these with RMA background correction. (Preferred analysis methods for Affymetrix GeneChips revealed by a wholly defined control dataset, 2005). What we observed here matches the conclusion from this paper. The intensities are normalized and all samples share similar pattern. However, the long tail around 0 intensity in the boxplot is weird, making variance of all samples stays large as un-normalized data. So PM only (no correction) is the best option in my case.

# compare different normalize.method  
par(mfrow = c(1, 2))  
boxplot(log2(exprs(affy\_data)), main = "Un-normalized")  
## the quantiles  
norm\_quan <- expresso(affy\_data, bgcorrect.method = "rma", summary.method = "avgdiff",   
 pmcorrect.method = "pmonly", normalize.method = "quantiles")

## background correction: rma   
## normalization: quantiles   
## PM/MM correction : pmonly   
## expression values: avgdiff   
## background correcting...done.  
## normalizing...done.  
## 54675 ids to be processed  
## | |  
## |####################|

boxplot(log2(exprs(norm\_quan)), main = "quantiles normalized avgdiff pmonly")



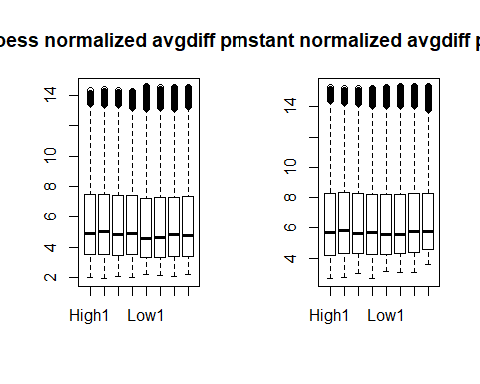
## the loess  
norm\_loess <- expresso(affy\_data, bgcorrect.method = "rma", summary.method = "avgdiff",   
 pmcorrect.method = "pmonly", normalize.method = "loess")

## background correction: rma   
## normalization: loess   
## PM/MM correction : pmonly   
## expression values: avgdiff   
## background correcting...done.  
## normalizing...Done with 1 vs 2 in iteration 1   
## Done with 1 vs 3 in iteration 1   
## Done with 1 vs 4 in iteration 1   
## Done with 1 vs 5 in iteration 1   
## Done with 1 vs 6 in iteration 1   
## Done with 1 vs 7 in iteration 1   
## Done with 1 vs 8 in iteration 1   
## Done with 2 vs 3 in iteration 1   
## Done with 2 vs 4 in iteration 1   
## Done with 2 vs 5 in iteration 1   
## Done with 2 vs 6 in iteration 1   
## Done with 2 vs 7 in iteration 1   
## Done with 2 vs 8 in iteration 1   
## Done with 3 vs 4 in iteration 1   
## Done with 3 vs 5 in iteration 1   
## Done with 3 vs 6 in iteration 1   
## Done with 3 vs 7 in iteration 1   
## Done with 3 vs 8 in iteration 1   
## Done with 4 vs 5 in iteration 1   
## Done with 4 vs 6 in iteration 1   
## Done with 4 vs 7 in iteration 1   
## Done with 4 vs 8 in iteration 1   
## Done with 5 vs 6 in iteration 1   
## Done with 5 vs 7 in iteration 1   
## Done with 5 vs 8 in iteration 1   
## Done with 6 vs 7 in iteration 1   
## Done with 6 vs 8 in iteration 1   
## Done with 7 vs 8 in iteration 1   
## 1 0.6054571   
## done.  
## 54675 ids to be processed  
## | |  
## |####################|

boxplot(log2(exprs(norm\_loess)), main = "loess normalized avgdiff pmonly")  
  
## constant  
norm\_cons <- expresso(affy\_data, bgcorrect.method = "rma", summary.method = "avgdiff",   
 pmcorrect.method = "pmonly", normalize.method = "constant")

## background correction: rma   
## normalization: constant   
## PM/MM correction : pmonly   
## expression values: avgdiff   
## background correcting...done.  
## normalizing...done.  
## 54675 ids to be processed  
## | |  
## |####################|

boxplot(log2(exprs(norm\_cons)), main = "constant normalized avgdiff pmonly")



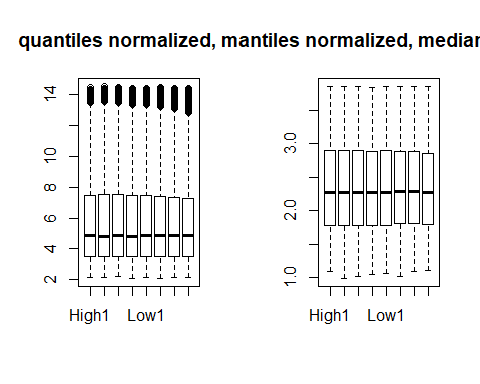
## try summary method mas  
quan\_mas <- expresso(affy\_data, bgcorrect.method = "rma", summary.method = "mas",   
 pmcorrect.method = "pmonly", normalize.method = "quantiles")

## background correction: rma   
## normalization: quantiles   
## PM/MM correction : pmonly   
## expression values: mas   
## background correcting...done.  
## normalizing...done.  
## 54675 ids to be processed  
## | |  
## |####################|

boxplot(log2(exprs(quan\_mas)), main = "quantiles normalized, mas")  
## medianpolish  
quan\_me <- expresso(affy\_data, bgcorrect.method = "rma", summary.method = "medianpolish",   
 pmcorrect.method = "pmonly", normalize.method = "quantiles")

## background correction: rma   
## normalization: quantiles   
## PM/MM correction : pmonly   
## expression values: medianpolish   
## background correcting...done.  
## normalizing...done.  
## 54675 ids to be processed  
## | |  
## |####################|

boxplot(log2(exprs(quan\_me)), main = "quantiles normalized, medianpolish")



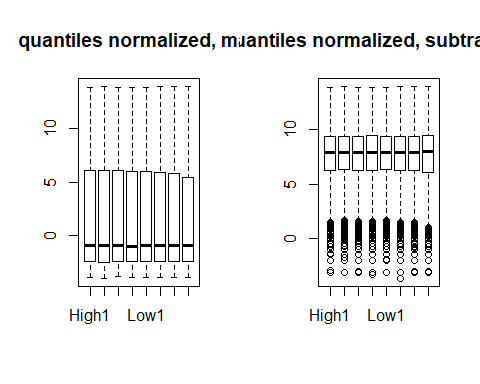
## try pmcorrect.method  
quan\_mas <- expresso(affy\_data, bgcorrect.method = "rma", summary.method = "avgdiff",   
 pmcorrect.method = "mas", normalize.method = "quantiles")

## background correction: rma   
## normalization: quantiles   
## PM/MM correction : mas   
## expression values: avgdiff   
## background correcting...done.  
## normalizing...done.  
## 54675 ids to be processed  
## | |  
## |####################|

boxplot(log2(exprs(quan\_mas)), main = "quantiles normalized, mas")  
  
quan\_sub <- expresso(affy\_data, bgcorrect.method = "rma", summary.method = "avgdiff",   
 pmcorrect.method = "subtractmm", normalize.method = "quantiles")

## background correction: rma   
## normalization: quantiles   
## PM/MM correction : subtractmm   
## expression values: avgdiff   
## background correcting...done.  
## normalizing...done.  
## 54675 ids to be processed  
## | |  
## |####################|

boxplot(log2(exprs(quan\_sub)), main = "quantiles normalized, subtractmm")



## d. Get present and absent calls Mas 5.0 using mas5calls()

The number of probesets have at least one present call in each of the two groups is 25726.

## present call  
pma\_call<- exprs(mas5calls(affy\_data))  
morethan\_one <- apply(pma\_call,1, function(x) {   
 High = sum(x[1:4]== "P")  
 Low = sum(x[5:8] == "P")  
 if(High >= 1 & Low >= 1)  
 {call = 1}  
 else{call = 0}  
return(call)  
})  
  
# this is the number of probesets pass the filter  
sum(morethan\_one)

## (e) If you think there are any problematic chip(s) based on your quality

I removed the “Low 4” sample. The .CEL image plot of “Low 4” sample shows there are contaminants in the raw image file. As I mentioned above, the big clot and black line in the image are unacceptable. The boxplot and density plot also show that there are amout of genes have very low intensity, which are abnormal. Particularly, the MA plot shows unusual pattern in the “Low 4” sample. All of these show that this sample is defective.

library(tidyverse)

## -- Attaching packages ---------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.0.0 v purrr 0.2.5  
## v tibble 1.4.2 v dplyr 0.7.6  
## v tidyr 0.8.1 v stringr 1.3.1  
## v readr 1.1.1 v forcats 0.3.0

## Warning: package 'tibble' was built under R version 3.4.4

## Warning: package 'tidyr' was built under R version 3.4.4

## Warning: package 'purrr' was built under R version 3.4.4

## Warning: package 'dplyr' was built under R version 3.4.4

## Warning: package 'stringr' was built under R version 3.4.4

## Warning: package 'forcats' was built under R version 3.4.4

## -- Conflicts ------------------------------------- tidyverse\_conflicts() --  
## x dplyr::combine() masks Biobase::combine(), BiocGenerics::combine()  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()  
## x ggplot2::Position() masks BiocGenerics::Position(), base::Position()  
## x readr::spec() masks genefilter::spec()

## present call  
pma\_call <- exprs(mas5calls(affy\_data))

## Getting probe level data...  
## Computing p-values  
## Making P/M/A Calls

morethan\_one <- apply(pma\_call, 1, function(x) {  
 High = sum(x[1:4] == "P")  
 Low = sum(x[5:8] == "P")  
 if (High >= 1 & Low >= 1) {  
 call = 1  
 } else {  
 call = 0  
 }  
 return(call)  
})  
  
# this is the number of probesets pass the filter  
sum(morethan\_one)

## [1] 25726

# get the reduced data the pheno Data  
pd\_r <- read.AnnotatedDataFrame("targets\_reduced.txt", header = TRUE,   
 row.names = 1, as.is = TRUE)  
# the affy data, READ CEL files into Affybatch  
affy\_data\_reduced <- ReadAffy(filenames = pData(pd\_r)$FileName,   
 phenoData = pd\_r, sampleNames = sampleNames(pd\_r))  
  
# normalize the data  
rma\_norm <- rma(affy\_data\_reduced, normalize = TRUE, background = TRUE)

## Background correcting  
## Normalizing  
## Calculating Expression

## let the reduced data pass the filter  
ex\_norm <- exprs(rma\_norm)  
  
pma\_norm\_call <- exprs(mas5calls(affy\_data\_reduced))

## Getting probe level data...  
## Computing p-values  
## Making P/M/A Calls

pre\_one <- apply(pma\_norm\_call, 1, function(x) {  
 High = sum(x[1:4] == "P")  
 Low = sum(x[5:7] == "P")  
 if (High >= 1 & Low >= 1) {  
 call = 1  
 } else {  
 call = 0  
 }  
 return(call)  
})  
  
# filter present call  
pma\_norm\_call <- data.frame(pma\_norm\_call)  
pma\_norm\_call$present <- pre\_one  
pma\_fil <- pma\_norm\_call %>% filter(present == 1)

## Warning: package 'bindrcpp' was built under R version 3.4.4

head(pma\_fil, 8)

## High1 High2 High3 High4 Low1 Low2 Low3 Low4 present  
## 1 P P P P P P P P 1  
## 2 P P P P P P P P 1  
## 3 P P P P P P P P 1  
## 4 P P P P P P P P 1  
## 5 P A P A M P P M 1  
## 6 P P P A A P M M 1  
## 7 P P P P P P A A 1  
## 8 P P P P P P P P 1

## filter intensity  
ex\_norm <- data.frame(ex\_norm)  
ex\_norm$present <- pre\_one  
ex\_fil <- ex\_norm %>% filter(present == 1)  
head(ex\_fil, 8)

## High1 High2 High3 High4 Low1 Low2 Low3  
## 1 9.938912 9.869950 9.903559 9.740580 11.388535 11.207691 11.314120  
## 2 9.313362 9.328511 9.324855 9.760381 9.189823 9.085477 9.175395  
## 3 6.457236 6.475070 6.423711 5.677630 5.818646 5.804252 5.647193  
## 4 7.327531 7.136042 7.165901 7.342722 7.279590 7.234276 7.430427  
## 5 5.445150 5.371184 5.632561 5.401077 5.487492 5.557352 5.501830  
## 6 4.013547 3.893491 4.158731 4.059701 3.922136 4.040567 4.077380  
## 7 3.960337 3.765239 3.800956 3.524164 3.599853 3.577387 3.537710  
## 8 6.431777 6.498224 6.480867 6.404712 6.195561 6.156480 6.245973  
## Low4 present  
## 1 11.350835 1  
## 2 9.178089 1  
## 3 5.515529 1  
## 4 7.374946 1  
## 5 5.428574 1  
## 6 4.043204 1  
## 7 3.502651 1  
## 8 6.143002 1