PB HLTH C240D/STAT C245D: Assignment #1

Rachael Phillips

September 28, 2017

Forensic Statistics: Duke Scandal Personalized Cancer Treatment Microarray Datasets

The purpose of this assignment is to get acquainted with basic and practical, yet crucial, statistical issues related to the analysis of high-throughput genomic data, by performing *exploratory data analysis* (EDA) and *quality assessment/control* (QA/QC) on microarray data from Potti et al. (2006).

Potti et al. (2006) propose an approach for predicting patient sensitivity to chemotherapeutic drugs based on *in vitro* drug sensitivity and gene expression measures for cell lines from the NCI60 panel. Coombes et al. (2007) and Baggerly and Coombes (2009) discuss their failure to reproduce the results in Potti et al. (2006), despite using the same data and software. They report a variety of simple, but serious, data processing errors that invalidate the conclusions of the study. More generally, they advocate the use of software such as **Sweave** to facilitate *reproducible* research.

Baggerly and Coombes (2009) summarize the approach in *Potti et al. (2006) as follows: Potti et al. (2006) introduced a method for combining microarray profiles of the NCI60 with drug sensitivity data to derive "signatures" of sensitivity to specific drugs, which they then used to predict patient response. In theory, the approach is straightforward:*

- *Using drug sensitivity data for a panel of cell lines, choose those that are most sensitive and most resistant to a drug.*
- *Using array profiles of the chosen cell lines, select the most differentially expressed genes.*
- *Using the selected genes, build a model that takes an array profile and returns a classification, and use this model to predict patient response.*

The R dataset **examiningDoxorubicinInDetail.RData**, posted on the class website, provides some of the data discussed in Baggerly and Coombes (2009) for the drug doxorubicin. Please consult http://projecteuclid.org/DPubS and http://bioinformatics.mdanderson.org/Supplements/ReproRsch-All/ for background on the various datasets and details on the pre-processing and analyses.

In particular, the matrix doxorubicinNCI60Scaled provides Affymetrix HG-U95Av2 expression measures for the training set of 22 cell lines from the NCI60 panel, classified as "Resistant" or "Sensitive" to doxorubicin. The column names are the NCI60 cell line names; their drug sensitivity status can be found in cellLinesUsed (use cellLinesUsed*doxorubicin*listPotti06CorrAug08 version). The row names are Affymetrix gene IDs. The data frame doxorubicin07Numbers provides another version of the same data for the training set of 22 cell lines, as well as Affymetrix HG-U133Av2 measures for a test set of 122 acute lymphoblastic leukemia (ALL) patients. The data frame doxorubicin07Info provides the training/test set membership of the samples and their drug sensitivity status, but not the actual names for the 22 cell lines comprising the training set and corresponding to the columns of doxorubicinNCI60Scaled. The row names are Affymetrix gene IDs; LocusLink IDs were used to match probes across the two Affymetrix platforms.

Note that, as indicated in Baggerly and Coombes (2009): *The data were log-transformed, the values for each row (gene) were centered and scaled to have mean zero and unit variance (separately for training and test data), the data were exponentiated to undo the log-transform, and the final results were rounded to two decimal places.* As microarray expression measures are based on fluorescence intensities (measured on a 16-bit scale), it may be appropriate to re-log-transform the data.

Question 1. Examine and prepare datasets.

Examine the different objects in the R dataset examiningDoxorubicinInDetail. Store the expression measures and the sample- and gene-level annotation metadata related to doxorubicinNCI60Scaled and doxorubicin07Numbers in objects of class *ExpressionSet* (Bioconductor R package Biobase).

```
# doxorubicinNCI60Scaled
assay_NCI60 <- doxorubicinNCI60Scaled
class(assay_NCI60)
```

```
## [1] "matrix"
```

```
dim(assay_NCI60)
```

```
## [1] 12625    22
```

```
status <- as.factor(ifelse(colnames(doxorubicinNCI60Scaled)
                           %in% cellLinesUsed$doxorubicin$listPotti06CorrAug08$Resistant,
                           "Resistant", "Sensitive"))
class(status)
```

```
## [1] "factor"
```

```
length(status)
```

```
## [1] 22
```

```
summary(status)
```

```
## Resistant Sensitive
##        12        10
```

```
pheno_NCI60 <- AnnotatedDataFrame(data.frame(status,
              row.names = colnames(assay_NCI60)))
summary(pheno_NCI60)
```

```
##             Length              Class               Mode
##                  1 AnnotatedDataFrame                 S4
```

```
dim(pheno_NCI60)
```

```
##    rowNames columnNames
##          22           1
```

```
head(pheno_NCI60)
```

```
## An object of class 'AnnotatedDataFrame'
##   rowNames: SF-539 SNB-75 ... MALME-3M (6 total)
##   varLabels: status
##   varMetadata: labelDescription
```

```
eSet_NCI60 <- ExpressionSet(assayData = assay_NCI60,
                            phenoData = pheno_NCI60)
head(eSet_NCI60)
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 1 features, 22 samples
##   element names: exprs
## protocolData: none
## phenoData
```

```
##   sampleNames: SF-539 SNB-75 ... CAKI-1 (22 total)
##   varLabels: status
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation:
```

```
class(exprs(eSet_NCI60))
```

```
## [1] "matrix"
```

```
head(exprs(eSet_NCI60)[,1:5])
```

```
##          SF-539 SNB-75 MDA-MB-435 NCI-H23  M14
## 36460_at   0.30   0.08       2.68    1.49 1.07
## 36461_at   3.13   1.20       6.18    3.55 0.34
## 36462_at   0.16   0.67       0.78    0.75 0.88
## 36463_at   1.43   0.83       0.94    2.37 0.66
## 36464_at   1.80   4.07       0.77    1.61 0.46
## 36465_at   2.25   1.15       0.04    0.93 0.18
```

```
# doxorubicin07Numbers
assay_dr07 <- as.matrix(doxorubicin07Numbers)
class(assay_dr07)
```

```
## [1] "matrix"
```

```
dim(assay_dr07)
```

```
## [1] 8958  144
```

```
pheno_dr07 <- AnnotatedDataFrame(doxorubicin07Info)
dim(pheno_dr07)
```

```
##    rowNames columnNames
##         144           2
```

```
head(pheno_dr07)
```

```
## An object of class 'AnnotatedDataFrame'
##   rowNames: Training1 Training2 ... Training6 (6 total)
##   varLabels: sampleGroup status
##   varMetadata: labelDescription
```

```
eSet_dr07 <- ExpressionSet(assayData = assay_dr07,
                           phenoData = pheno_dr07)
head(eSet_dr07)
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 1 features, 144 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: Training1 Training2 ... Test122 (144 total)
##   varLabels: sampleGroup status
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation:
```

```
head(exprs(eSet_dr07)[,1:5])
```

```
##          Training1 Training2 Training3 Training4 Training5
## 35753_at      1.18      1.12      3.46      0.65      3.07
## 36138_at      1.75      4.02      0.43      0.31      0.76
## 41765_at      0.13      0.35      1.13      1.14      0.84
## 35298_at      0.19      0.42      0.52      0.52      1.32
## 38974_at      0.63      0.98      1.22      2.02      2.22
## 41194_at      1.01      0.49      1.11      2.16      1.43
```

Question 2. QA/QC for training set.

Reconcile the training data in doxorubicinNCI60Scaled and doxorubicin07Numbers, i.e., match genes and samples and compare the expression measures and the sensitivity status assigned to the cell lines in the two datasets.

```
# matching the affy IDs
affy_match <- rownames(eSet_NCI60) %in% rownames(eSet_dr07)
# making a combined ExpressionSet
eSet_combined <- BiocGenerics::combine(eSet_NCI60[affy_match,],
                                       eSet_dr07[,1:22])
head(eSet_combined)
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 1 features, 44 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: SF-539 SNB-75 ... Training22 (44 total)
##   varLabels: status sampleGroup
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation:
```

```
# we "tag" the samples with their corresponding sensitivity status
# so we don't have to pull this information from the pheno table
# and we don't have to color by these groups for EDA
name_status <- ifelse(eSet_combined$status == "Sensitive",
                      paste("S", colnames(eSet_combined), sep = "_"),
                      paste("R", colnames(eSet_combined), sep = "_"))
# just making sure we didn't mess up!
sample_n(data.frame(status = eSet_combined$status,
                    sample = rownames(pData(eSet_combined)),
                    combined = name_status), 5)
```

```
##       status     sample     combined
## 23 Resistant  Training1  R_Training1
## 41 Sensitive Training19 S_Training19
## 2  Sensitive     SNB-75     S_SNB-75
## 40 Sensitive Training18 S_Training18
## 27 Resistant  Training5  R_Training5
```

```r
# now we can modify the ExpressionSet
colnames(eSet_combined) <- name_status

# EDA
head(exprs(eSet_combined)[,1:5])
```

```
##            S_SF-539 S_SNB-75 S_MDA-MB-435 S_NCI-H23 S_M14
## 36460_at       0.30     0.08         2.68      1.49  1.07
## 36461_at       3.13     1.20         6.18      3.55  0.34
## 36462_at       0.16     0.67         0.78      0.75  0.88
## 36463_at       1.43     0.83         0.94      2.37  0.66
## 36464_at       1.80     4.07         0.77      1.61  0.46
## 36471_f_at     2.24     1.32         1.75      0.73  0.55
```

```r
head(exprs(eSet_combined)[,23:27])
```

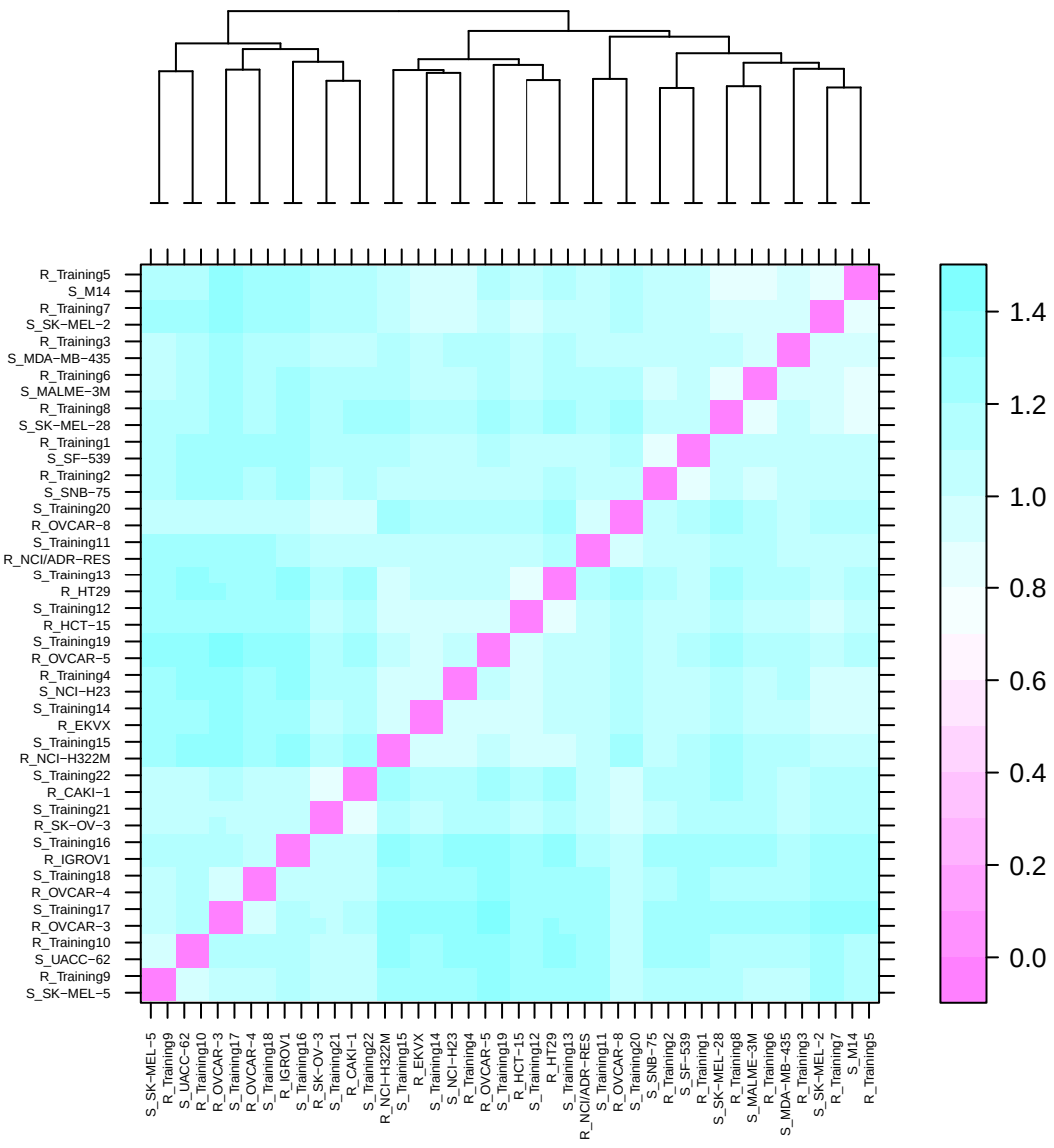
```
##            R_Training1 R_Training2 R_Training3 R_Training4 R_Training5
## 36460_at          0.30        0.08        2.68        1.49        1.07
## 36461_at          3.13        1.20        6.18        3.55        0.34
## 36462_at          0.16        0.67        0.78        0.75        0.88
## 36463_at          1.43        0.83        0.94        2.37        0.66
## 36464_at          1.80        4.07        0.77        1.61        0.46
## 36471_f_at        2.24        1.32        1.75        0.73        0.55
```

```r
# it seems that column i corresponds to column i+22 based
# on the expression measures

# however these samples have different status labels, that's odd
data.frame(NCI60_status = pData(eSet_combined)[1:5,-2],
           dr07_status= pData(eSet_combined)[23:27,-2])
```

```
##   NCI60_status dr07_status
## 1    Sensitive   Resistant
## 2    Sensitive   Resistant
## 3    Sensitive   Resistant
## 4    Sensitive   Resistant
## 5    Sensitive   Resistant
```

```r
# let's investigate further

# We compare the expression measures and sensitiviy status assigned

# we calculate a measure of sample dissimilarity via pairwise
# differences that can be used for heirarchical cluster analysis

# By default, dist2 calculates the mean of the absolute differences
# between pairs of samples.
dd <- dist2(log(exprs(eSet_combined)))
dd.row <- as.dendrogram(hclust(as.dist(dd)))
# ordering the similar samples next to eachother
row.ord <- order.dendrogram(dd.row)
legend <- list(top = list(fun = dendrogramGrob,
                          args = list(x = dd.row, side = "top")))
lp <- levelplot(dd[row.ord, row.ord],
                scales = list(x = list(rot = 90, cex=.45),y=list(cex=.45)),
                main = "Between-Array Distance Plot: Comparing the Expression
```

```
                Measures and Sensitivty Status Assigned to the Cell Lines
                in the Two Datasets", xlab = "", ylab = "",
                legend = legend)

plot(lp)
```



This graphs displays the human error perfectly. We can see that there are indeed matches between the datasets because, for all 22 samples there is a bright pink box surrounding two samples, i.e. 22 pairs with zero between-array dissimilarity. If we look close enough at this plot we see that, for each pair, the sensitivity status is reversed!

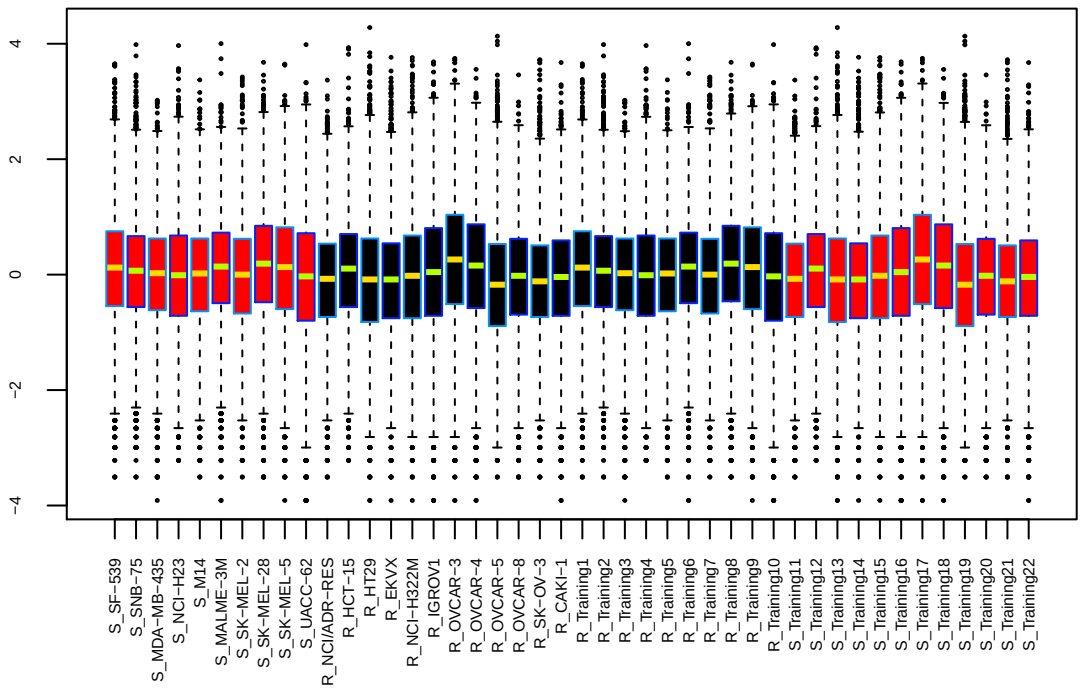
```
order_by(as.vector(row.ord), colnames(eSet_combined))
```

```
##  [1] "S_SK-MEL-5"    "R_Training9"   "S_UACC-62"     "R_Training10"
```

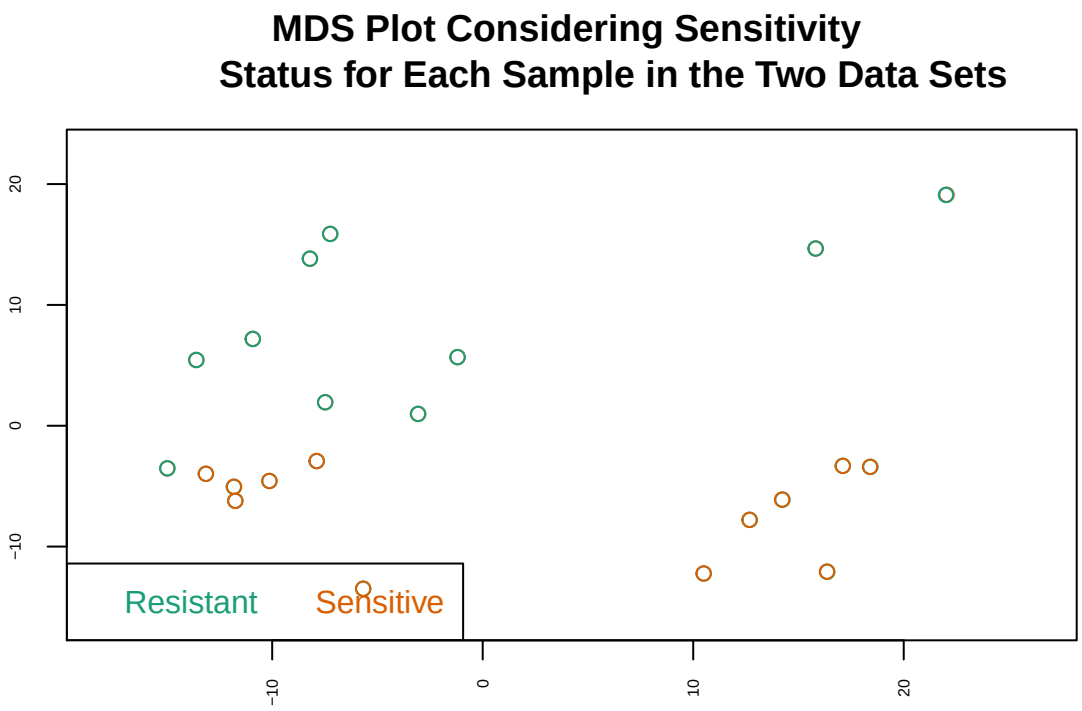
```
##  [5] "R_OVCAR-3"     "S_Training17"  "R_OVCAR-4"     "S_Training18"
##  [9] "R_IGROV1"      "S_Training16"  "R_SK-OV-3"     "S_Training21"
## [13] "R_CAKI-1"      "S_Training22"  "R_NCI-H322M"   "S_Training15"
## [17] "R_EKVX"        "S_Training14"  "S_NCI-H23"     "R_Training4"
## [21] "R_OVCAR-5"     "S_Training19"  "R_HCT-15"      "S_Training12"
## [25] "R_HT29"        "S_Training13"  "R_NCI/ADR-RES" "S_Training11"
## [29] "R_OVCAR-8"     "S_Training20"  "S_SNB-75"      "R_Training2"
## [33] "S_SF-539"      "R_Training1"   "S_SK-MEL-28"   "R_Training8"
## [37] "S_MALME-3M"    "R_Training6"   "S_MDA-MB-435"  "R_Training3"
## [41] "S_SK-MEL-2"    "R_Training7"   "S_M14"         "R_Training5"
```

Here we can see a larger view of the pairs that correspond to a match in samples across these datasets. For example, "S_SK-MEL-5" and "R_Training9" are a match and "S_UACC-62" and "R_Training10" are a match. This visual shows that, indeed, every pair of matches has the sensitivity status reversed. We also include a few additional QC/QA plots for good measure.

```
par(cex.axis=.5, las=3)
boxplot(log(exprs(eSet_combined)),col=eSet_combined$status,
        boxcol=c("#0092FFFF", "#0024FFFF"),
        medcol=c("#FFDB00FF", "#B6FF00FF"),
        cex=.2, main = "Boxplots of Expression Measures
        for Each Sample in the Two Data Sets")
```




```
# Euclidean distance is calculated between samples using the
# 1000 most variable exprs positions. These distances are
# then projected into a 2-d plane using classical
# multidimensional scaling transformation.
mdsPlot(log(exprs(eSet_combined)), sampGroups = eSet_combined$status,
main = "MDS Plot Considering Sensitivity
        Status for Each Sample in the Two Data Sets")
```



Boxplots are useful to highlight aberrent samples, outliers, and vizualize the distribution of the expression measures across samples and sensitivity status. We do not see any problematic samples according to the boxplot. Multi-dimensinal scaling plots give an overview of similarities and differences between samples so this is another useful QC visual.

Question 3. QA/QC for test set.

Consider now the test data in doxorubicin07Numbers. Is there anything unusual with the samples and their assigned sensitivity statuses?

```
# Consider now the test data in doxorubicin07Numbers:
eSet_dr07Test <- eSet_dr07[,23:144]

# like before,
# we "tag" the samples with their corresponding sensitivity status
# so we don't have to pull this information from the pheno table
# and we don't have to color by these groups for EDA
name_status <- ifelse(eSet_dr07Test$status == "Sensitive",
                      paste("S", colnames(eSet_dr07Test), sep = "_"),
                      paste("R", colnames(eSet_dr07Test), sep = "_"))
# making sure we didn't mess up
sample_n(data.frame(status = eSet_dr07Test$status,
                    sample = rownames(pData(eSet_dr07Test)),
                    combined = name_status), 5)
```

```
##       status sample combined
## 21 Sensitive Test21 S_Test21
## 87 Resistant Test87 R_Test87
## 27 Sensitive Test27 S_Test27
```

```
## 36 Resistant Test36 R_Test36
## 3  Sensitive  Test3  S_Test3
```

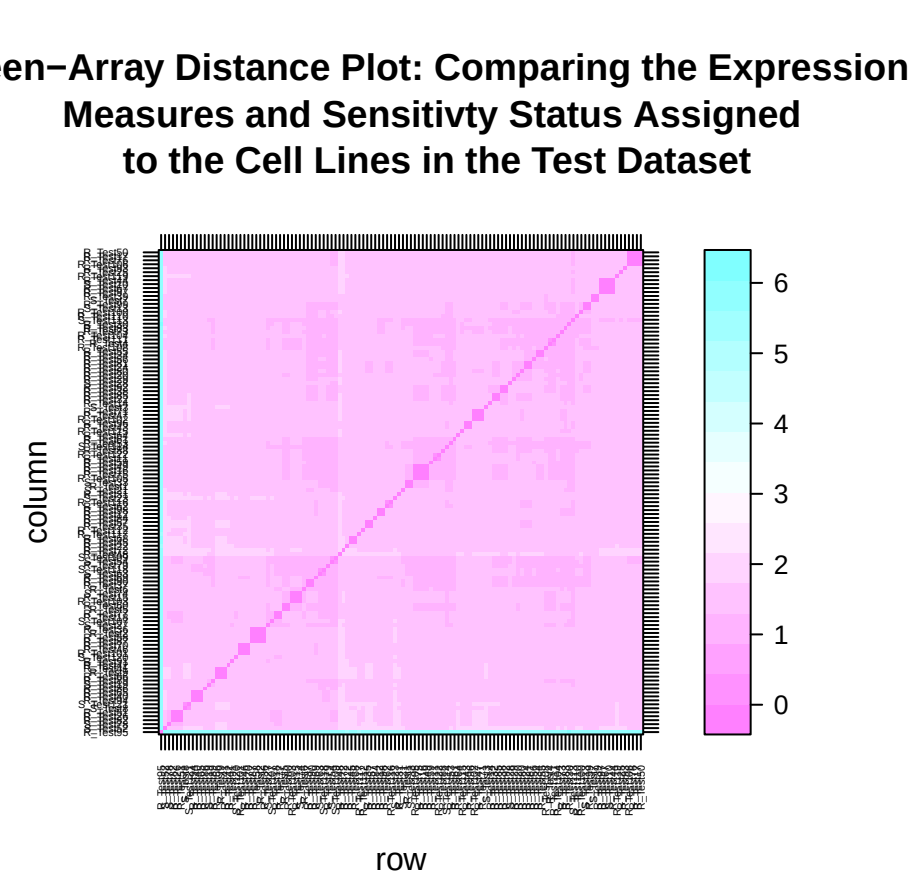
```
# now we can modify the ExpressionSet
colnames(eSet_dr07Test) <- name_status

# Next, we compare the expression measures and sensitiviy status assigned
# We calculate a measure of dissimilarity via pairwise
# differences that can be used for heirarchical cluster analysis

# By default, dist2 calculates the mean of the absolute differences
# between pairs of samples.
dd <- dist2(log2(exprs(eSet_dr07Test)))
sum(is.infinite(as.dist(dd)))
```

```
## [1] 5026
```

```
# we cannot log transform the data as the dendrogram fails for
# infinite values
dd <- dist2(exprs(eSet_dr07Test))
sum(is.infinite(as.dist(dd)))
```

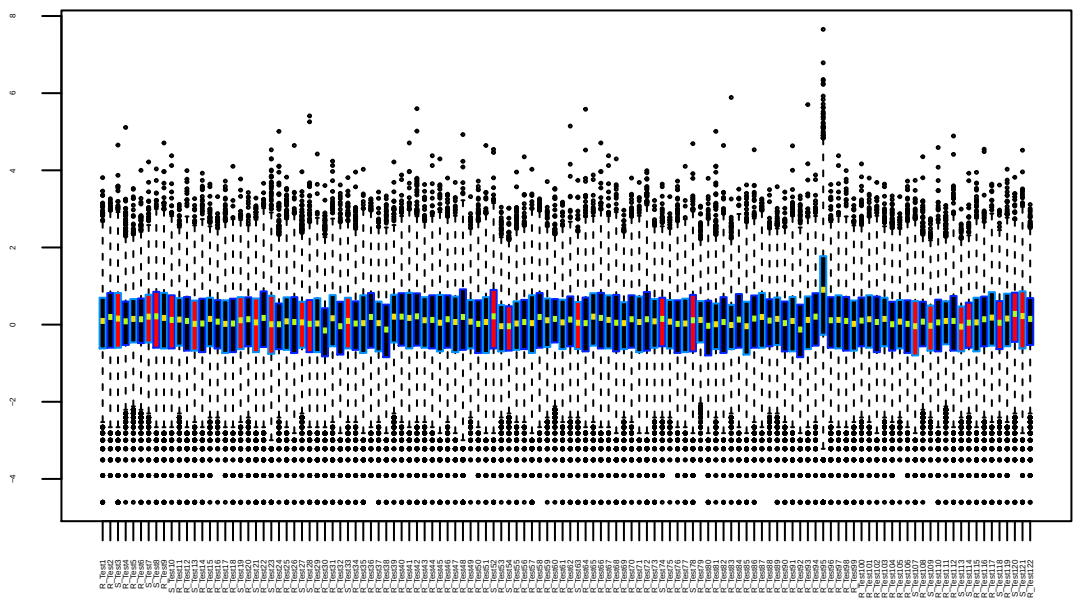
```
## [1] 0
```

```
dd.row <- as.dendrogram(hclust(as.dist(dd)))
# ordering the similar samples next to eachother
row.ord <- order.dendrogram(dd.row)
legend <- list(top = list(fun = dendrogramGrob,
                          args = list(x = dd.row, side = "top")))
lp <- levelplot(dd[row.ord, row.ord], scales = list(x = list(rot = 90, cex=.3),
      xlab = "", ylab= "", legend = legend, y=list(cex=.3)),
main = "Between-Array Distance Plot: Comparing the Expression
      Measures and Sensitivty Status Assigned
      to the Cell Lines in the Test Dataset")

plot(lp)
```

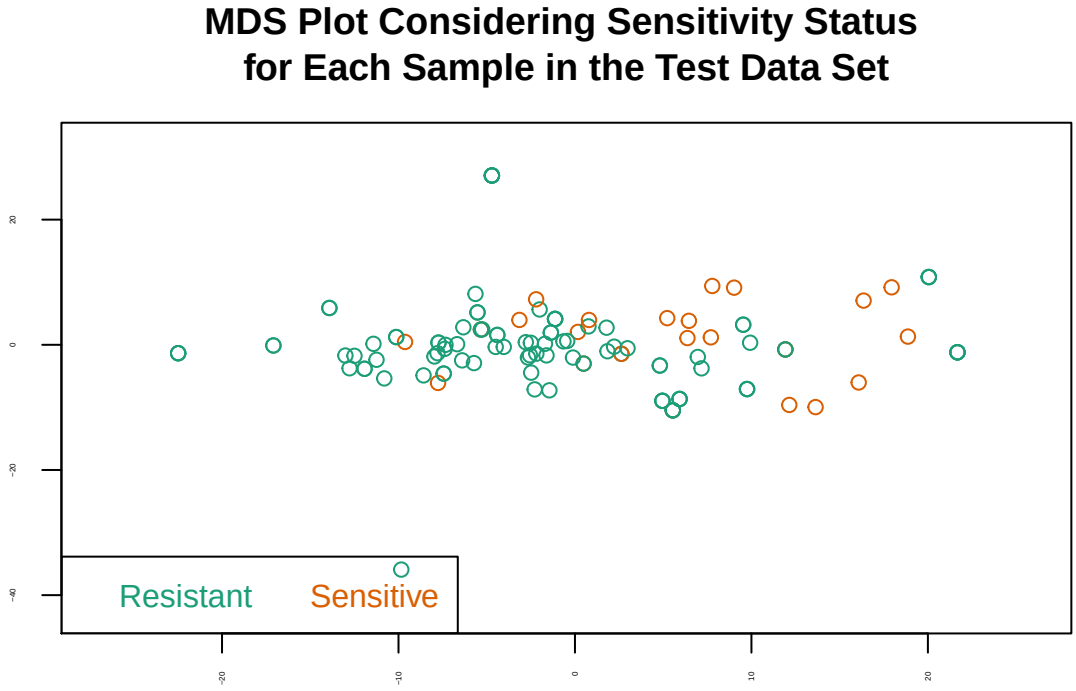


There is something to notice on the Between Array Distance Plot. The brightest pink color corresponding simply to zero between array dissimilarity should be zero when a sample is compared against itself. This is why the diagonal line is the brightest pink color. However, we see that there are some places on the diagonal that are "bulged". That is, there are two samples with the zero between-array dissimilarity. This seems very odd, we would expect, at the least, subtle between sample variations. Thus, we believe that some samples were duplicated.

```
par(cex.axis=.25, las=3)
boxplot(log(exprs(eSet_dr07Test)),col=eSet_dr07Test$status,
        boxcol=c("#0092FFFF", "#0024FFFF"),
        medcol=c("#FFDB00FF", "#B6FF00FF"),
        cex=.2, main = "Boxplots of Expression Measures
        for Each Sample in the Test Data Set")
```




```
# Euclidean distance is calculated between samples using the
# 1000 most variable exprs positions. These distances are
# then projected into a 2-d plane using classical
# multidimensional scaling transformation.
mdsPlot(log(exprs(eSet_dr07Test)), sampGroups = eSet_dr07Test$status,
main = "MDS Plot Considering Sensitivity Status
for Each Sample in the Test Data Set")
```



According to the boxplots, we see that one sample is problematic. There is a large difference in both location

and spread between this problematic sample and the other samples. We should consider removing this sample if we use this data for the normalization. Accoring to the MDS plot, there is not a clear difference between 'Sensitive' and 'Resistant' samples.

Question 4. Differential expression analysis for training set.

The goal of this question is to derive, as in Potti et al. (2006), a "signature" of sensitivity to doxorubicin, which could be used eventually to predict patient response to the drug. This involves identifying genes that are *differentially expressed* (DE) between "Resistant" and "Sensitive" cell lines based on the training set of microarray expression measures for the 22 NCI60 cell lines.

Use the expression measures stored in doxorubicinNCI60Scaled and the drug sensitivity status given by cellLinesUsed*doxorubicin*listPotti06CorrAug08.

a) Between-sample normalization.

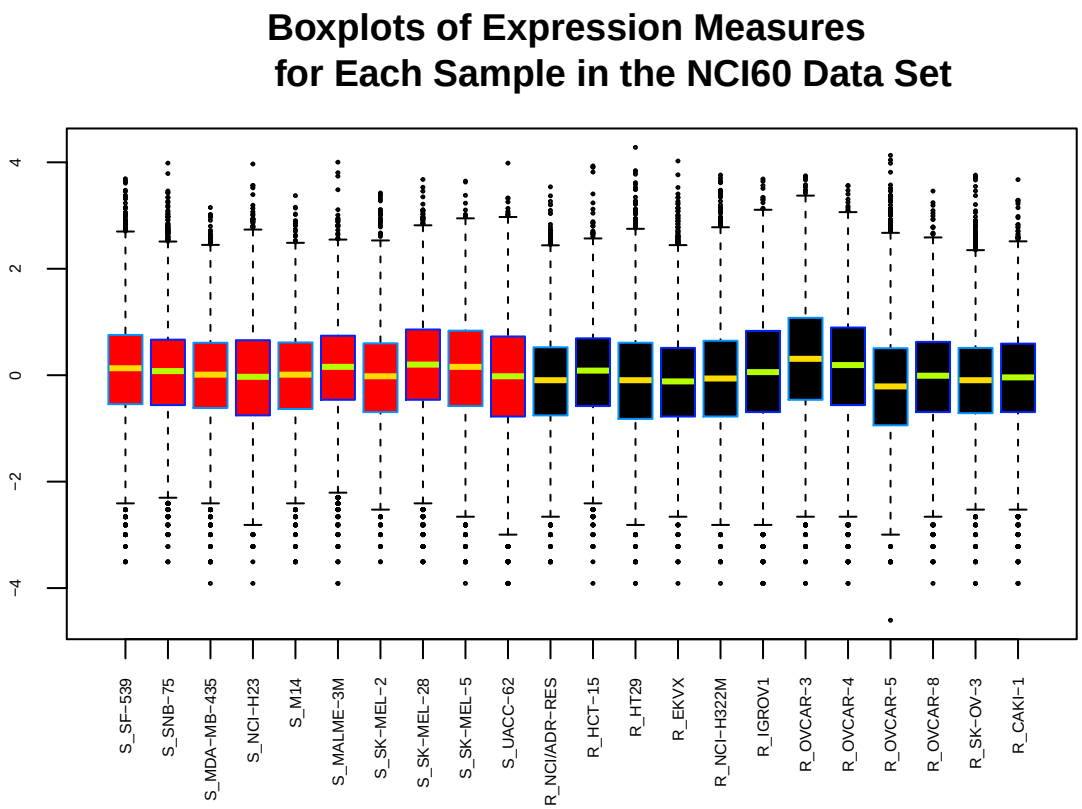
In order to derive expression measures and compare these measures between samples and/or genomic regions of interest (ROI), one first needs to *normalize* the fluorescence intensities to adjust for a variety of sample-level and gene-level technical effects, such as, library preparation/hybridization/scanning effets and nucleotide composition effects (e.g., GC-content). Normalization is essential to ensure that observed differences in expression measures between samples and/or ROI are truly due to differential expression and not technical artifacts. For the purpose of this assignment, it is sufficient to perform between-sample normalization using standard methods for Affymetrix (one-channel) microarrays, e.g., full-quantile and loess procedures implemented in Bioconductor R packages such as affy and limma. Provide and comment on numerical and graphical summaries of the data that suggest the need for normalization. Perform between-sample normalization and comment on the results.

```
# like before,
# we will "tag" the samples with their corresponding sensitivity status
# so we don't have to pull this information from the pheno table
# and we don't have to color by these groups for EDA
name_status <- ifelse(eSet_NCI60$status == "Sensitive",
                      paste("S", colnames(eSet_NCI60), sep = "_"),
                      paste("R", colnames(eSet_NCI60), sep = "_"))
# making sure we didn't mess up
sample_n(data.frame(status = eSet_NCI60$status,
                    sample = rownames(pData(eSet_NCI60)),
                    combined = name_status), 5)
```

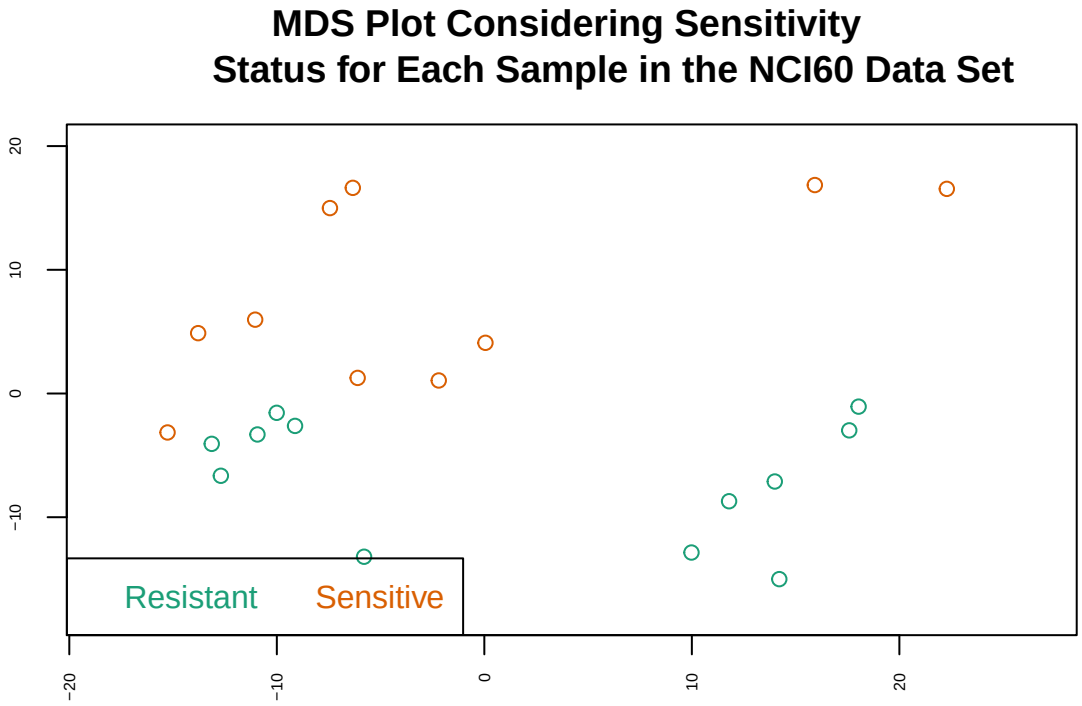
```
##       status  sample  combined
## 12 Resistant  HCT-15  R_HCT-15
## 17 Resistant OVCAR-3 R_OVCAR-3
## 14 Resistant    EKVX    R_EKVX
## 1  Sensitive  SF-539  S_SF-539
## 18 Resistant OVCAR-4 R_OVCAR-4
```

```
# now we can modify the ExpressionSet
colnames(eSet_NCI60) <- name_status

# Provide and comment on numerical and graphical summaries of the data
# that suggest the need for normalization.

par(cex.axis=.5, las=3)
```

```
boxplot(log(exprs(eSet_NCI60)),col=eSet_NCI60$status,
        boxcol=c("#0092FFFF", "#0024FFFF"),
        medcol=c("#FFDB00FF", "#B6FF00FF"),
        cex=.2, main = "Boxplots of Expression Measures
        for Each Sample in the NCI60 Data Set")
```

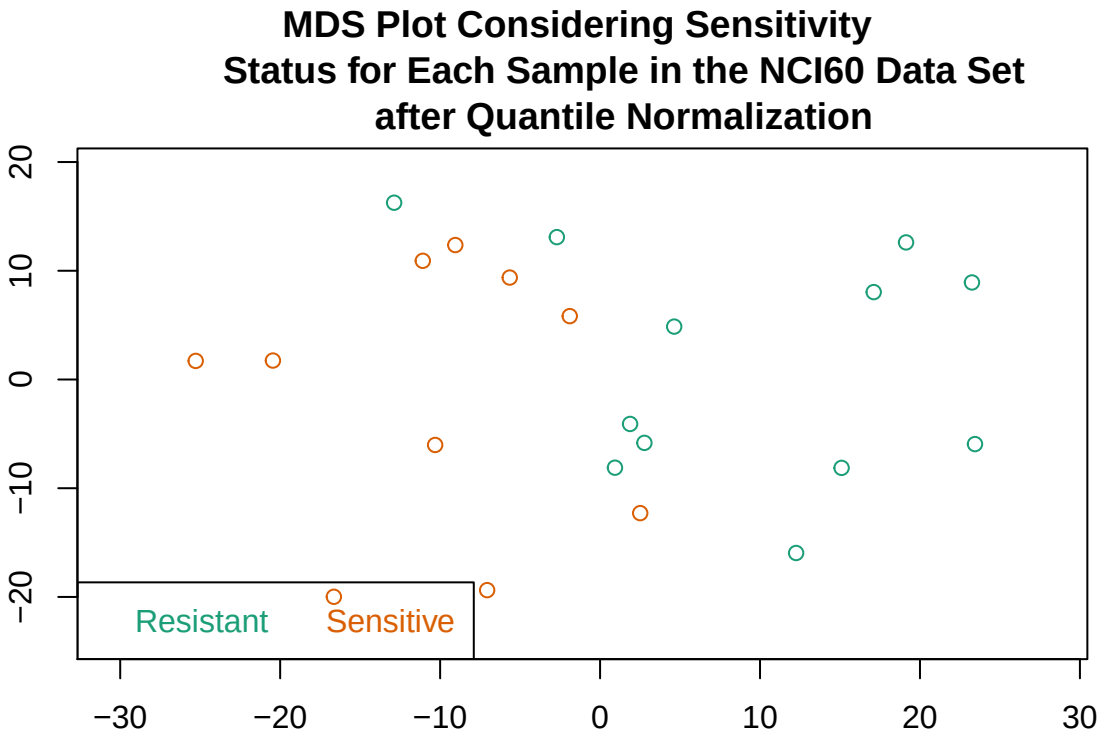



```
# Euclidean distance is calculated between samples using the
# 1000 most variable exprs positions. These distances are
# then projected into a 2-d plane using classical
# multidimensional scaling transformation.
mdsPlot(log(exprs(eSet_NCI60)), sampGroups = eSet_NCI60$status,
main = "MDS Plot Considering Sensitivity
        Status for Each Sample in the NCI60 Data Set")
```

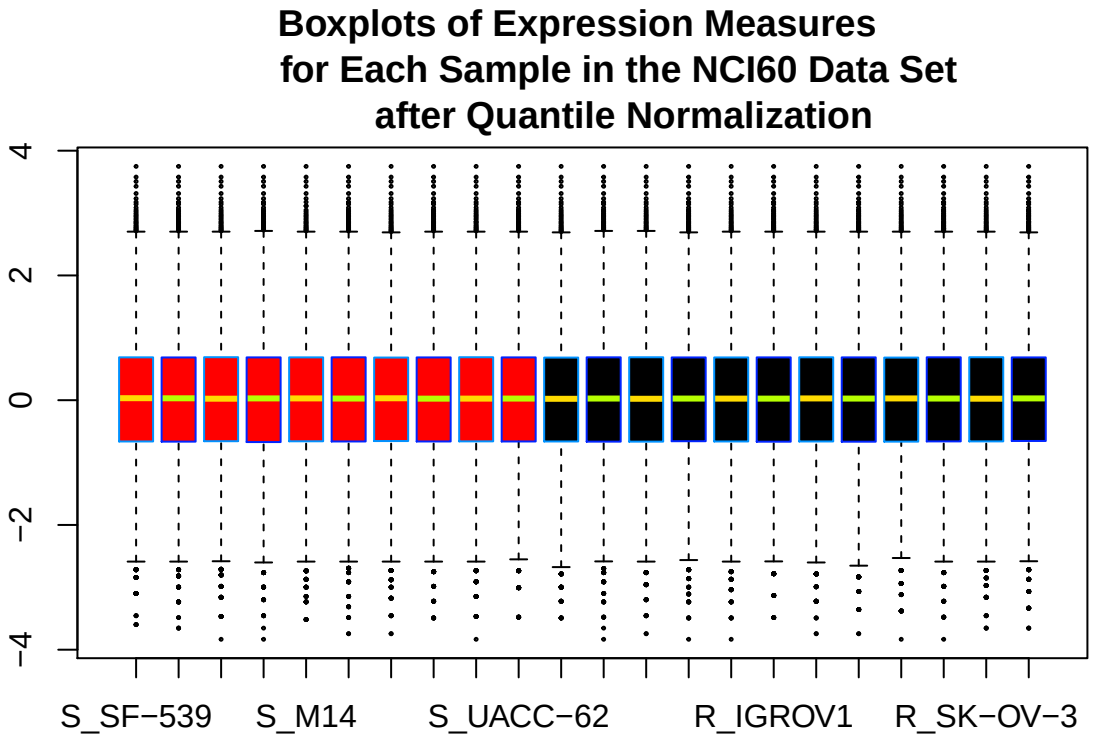


The MDS plot shows that the differences between samples are not completely clustered by the sensitivity status but we do see some clustering by sensitivity status. The boxplots show differences in the ditributions between samples and not between status groups. We are interested in identifying genes that are differentially expressed in the "Resistant" and "Sensitive" cell lines. To ensure that observed differences in expression measures between samples and/or regions of interest are truly due to differential expression between these status groups and not technical artifacts we need to normalize the data.

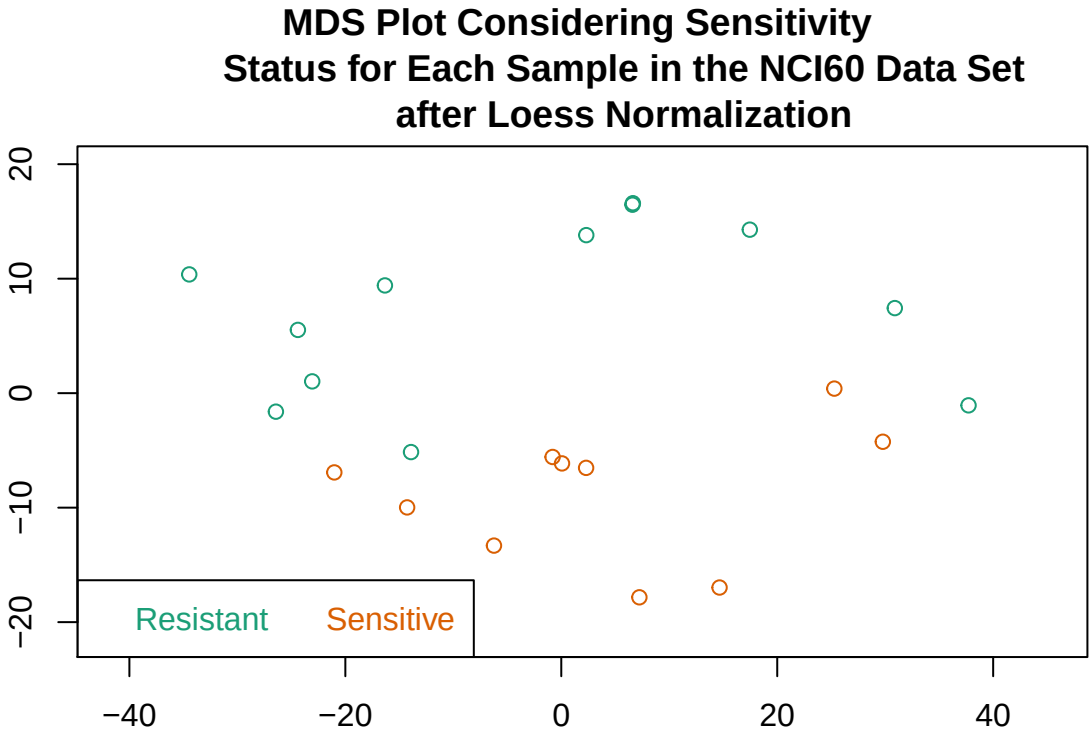
```
# Perform between-sample normalization and comment on the results.

# Quantile normalization forces the entire empirical
# distribution of each column to be identical.
exprs_quantile <- normalizeBetweenArrays(object = log(exprs(eSet_NCI60)),
                                         method = "quantile")

mdsPlot(exprs_quantile, sampGroups = eSet_NCI60$status,
main = "MDS Plot Considering Sensitivity
        Status for Each Sample in the NCI60 Data Set
        after Quantile Normalization")
```



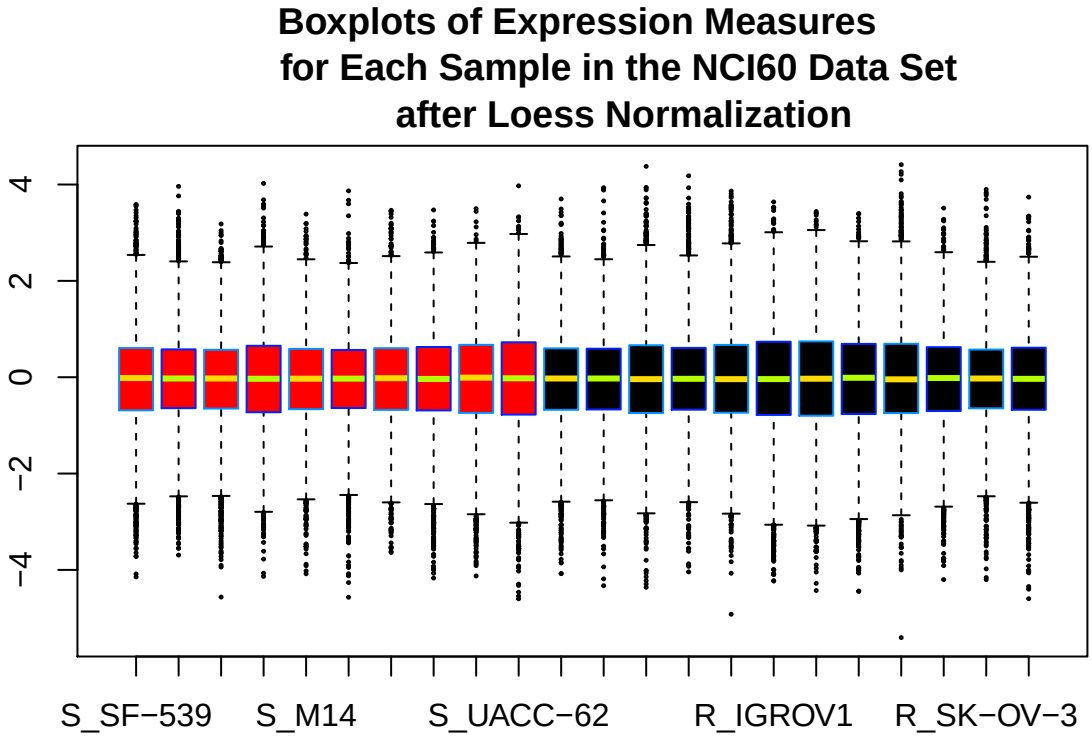

```
boxplot(exprs_quantile,col=eSet_NCI60$status,
        boxcol=c("#0092FFFF", "#0024FFFF"),
        medcol=c("#FFDB00FF", "#B6FF00FF"),
        cex=.2, main = "Boxplots of Expression Measures
        for Each Sample in the NCI60 Data Set
        after Quantile Normalization")
```



```
# Cyclic loess normalization applies loess normalization to all
# possible pairs of arrays, usually cycling through all pairs
# several times. Cyclic loess is slower than quantile, but allows
# probe-wise weights and is more robust to unbalanced differential
# expression

exprs_loess <- normalizeBetweenArrays(object = log(exprs(eSet_NCI60)),
                                      method = "cyclicloess")

mdsPlot(exprs_loess, sampGroups = eSet_NCI60$status,
main = "MDS Plot Considering Sensitivity
        Status for Each Sample in the NCI60 Data Set
        after Loess Normalization")
```




```
boxplot(exprs_loess,col=eSet_NCI60$status,
        boxcol=c("#0092FFFF", "#0024FFFF"),
        medcol=c("#FFDB00FF", "#B6FF00FF"),
        cex=.2, main = "Boxplots of Expression Measures
        for Each Sample in the NCI60 Data Set
        after Loess Normalization")
```

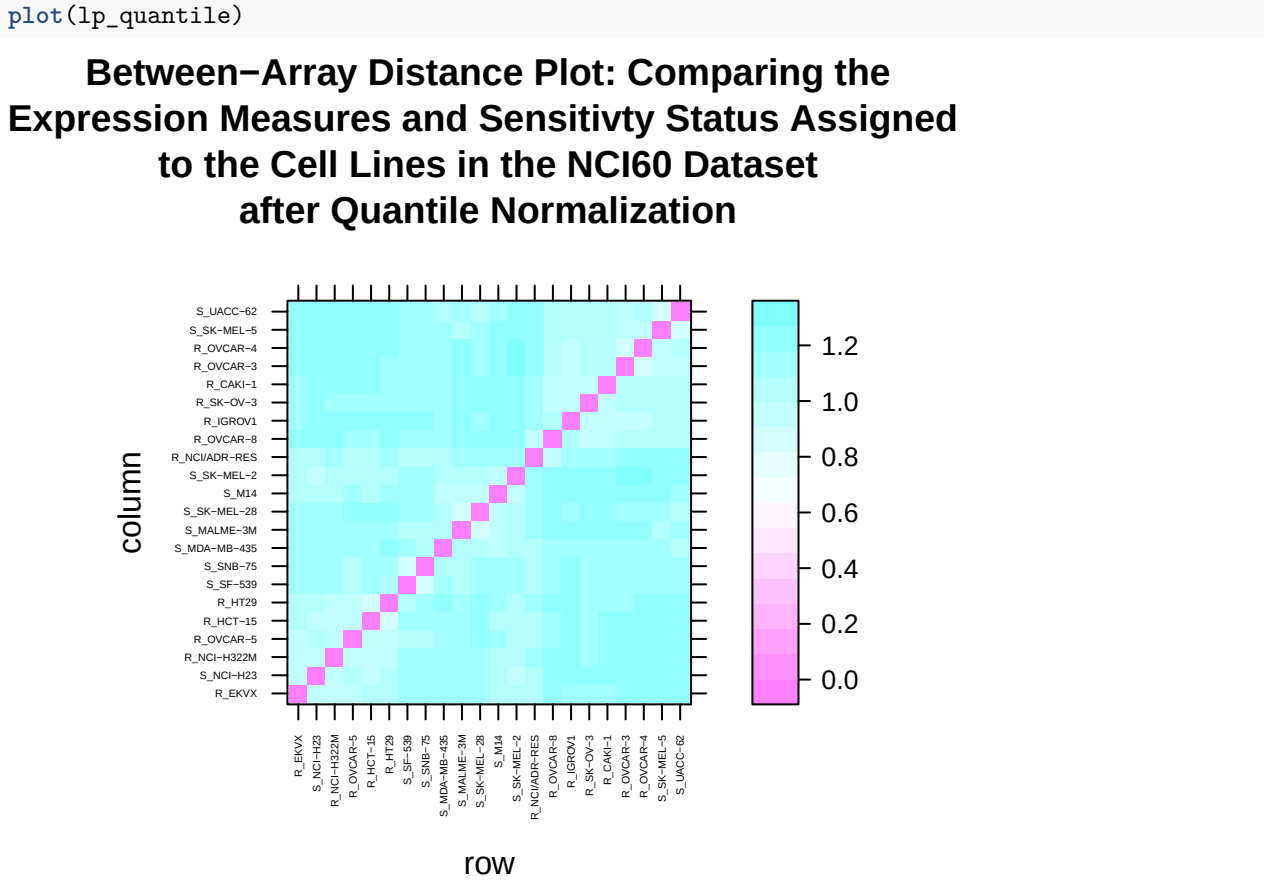


We consider both Quantile and Loess normalization and we will proceed with the remainder of the analysis without removing a normalization. We do notice the Loess normalization has more separation in the MDS plot and the boxplots under the Quantile normalization look very symmetric with each other. According to *A Comparison of Normalization Methods for High Density Oligonucleotide Array Data Based on Variance and Bias* (Bolstad, 2003), "The contrast and cyclic loess algorithms are modifications of an accepted method of normalization. The quantile method has performed favorably, both in terms of speed and when using our variance and bias criteria, and therefore should be used in preference to the other methods." Ideally, we would like to be able to compare these normalizations statistically in addition to this visual comparison. We do not feel comfortable ruling one of them out based on visual QC/QA performance or based on comparisions that did not use this data. Thus, we procede to cluster analysis for both of these normalizations.

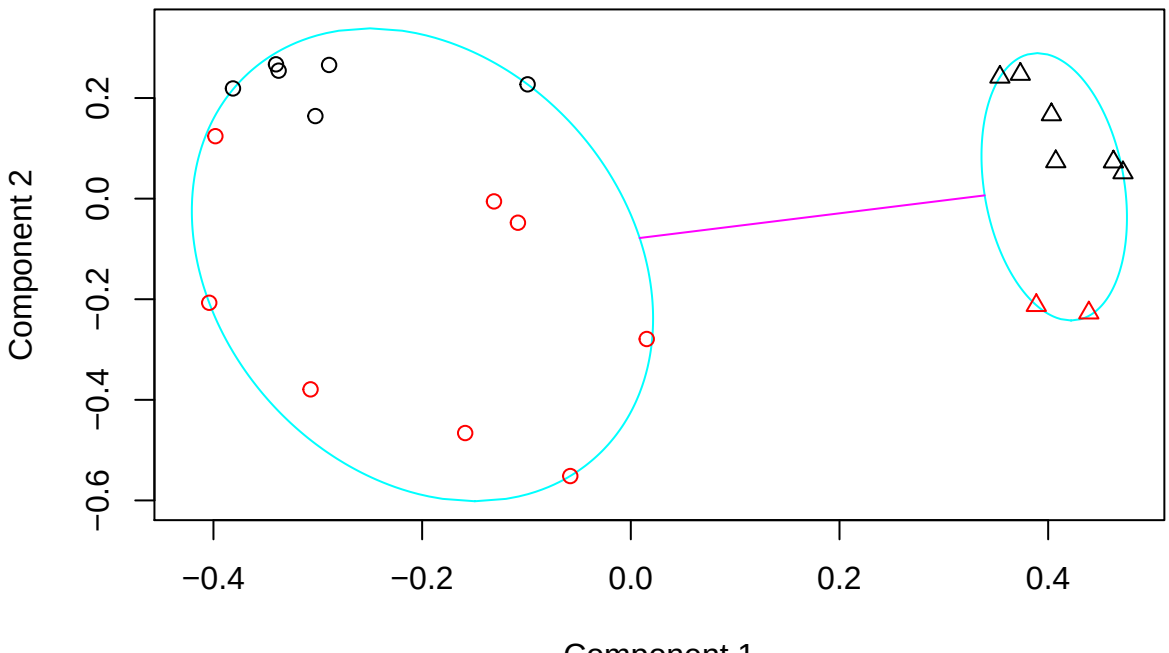
b) Cluster analysis. Apply *dimensionality reduction* and *clustering* (both hierarchical and partitioning) methods to the training set samples. Comment on the results and, in particular, relate the clustering to drug sensitivity.

```
# Quantile Normalization

# Heirarchical Clustering
# a graphical representation of a matrix of distances
# a dendrogram, or tree is a plot where the objects are joined
# together in a hierarchical fashion from the closest, that is most
# similar, to the furthest apart, that is the most different
d_quantile <- dist2(exprs_quantile)
dend_quantile <- as.dendrogram(hclust(as.dist(d_quantile)))
row.ord_quantile <- order.dendrogram(dend_quantile)
legend_quantile <- list(top = list(fun = dendrogramGrob,
                        args = list(x = dend_quantile, side = "top")))
lp_quantile <- levelplot(d_quantile[row.ord_quantile, row.ord_quantile],
scales = list(x = list(rot = 90, cex=.3), xlab = "",
              ylab= "", legend = legend, y =list(cex=.3)),
main = "Between-Array Distance Plot: Comparing the
```

```
Expression Measures and Sensitivty Status Assigned
to the Cell Lines in the NCI60 Dataset
after Quantile Normalization")
plot(lp_quantile)
```

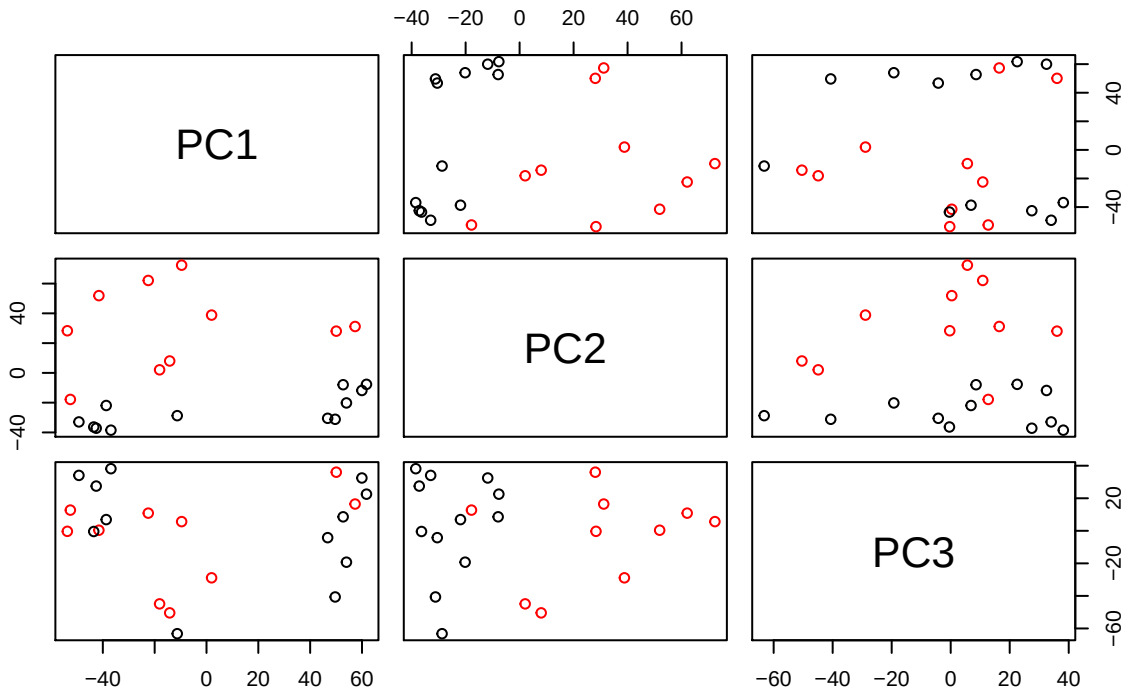



```
# Partitioning Around Medoids (PAM) Clustering
# The algorithm is intended to find a sequence of medoids
# (the distance between two groups as the distance between their centroids
# (center of gravity or vector average) that are centrally located in
# clusters. The goal of the algorithm
# is to minimize the average dissimilarity of objects to their closest
# selected object. Equivalently, we can minimize the sum of the
# dissimilarities between object and their closest selected object.
pq <- cluster::pam(as.dist(d_quantile),k=2,diss=TRUE)
cluster::clusplot(pq,col.p=eSet_NCI60$status,
         main="PAM Clustering of NCI60 Dataset after Quantile Normalization")
```

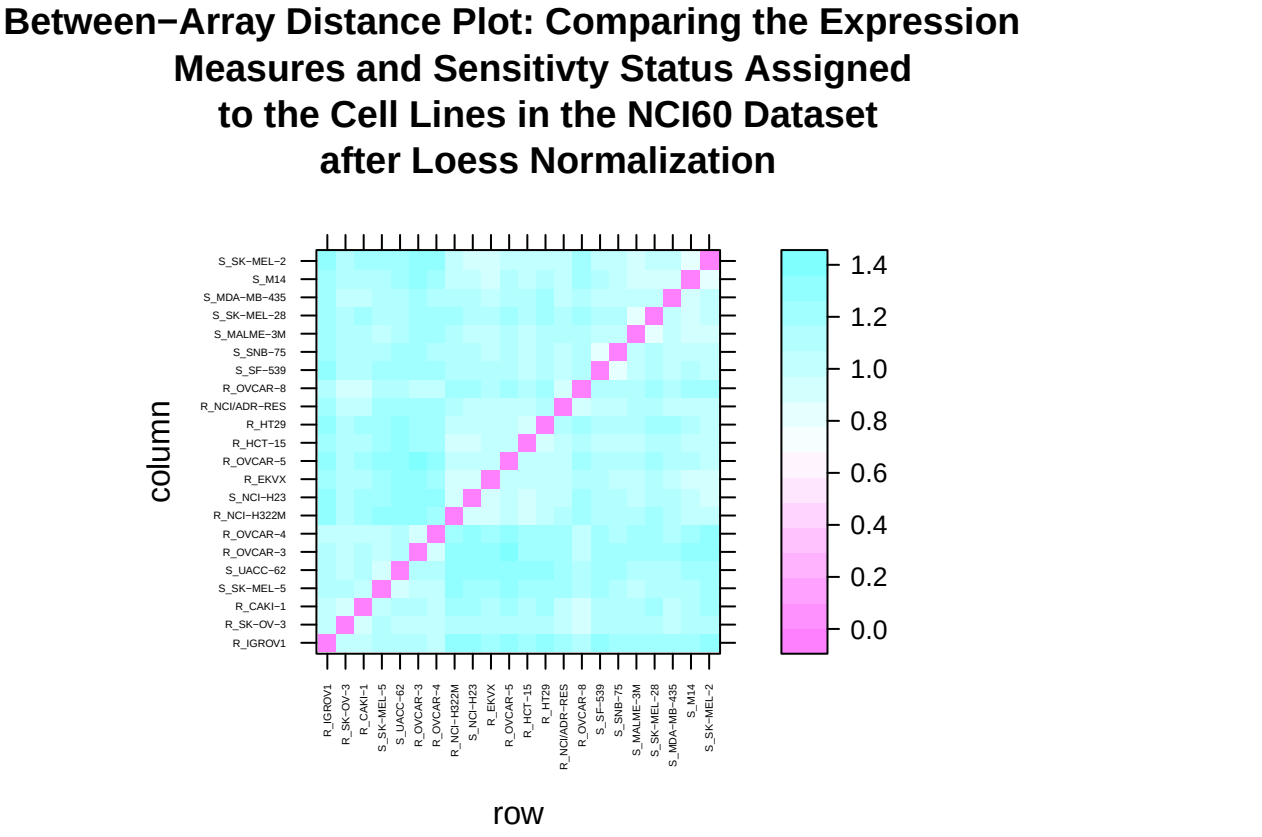


These two components explain 28.9 % of the point variability.

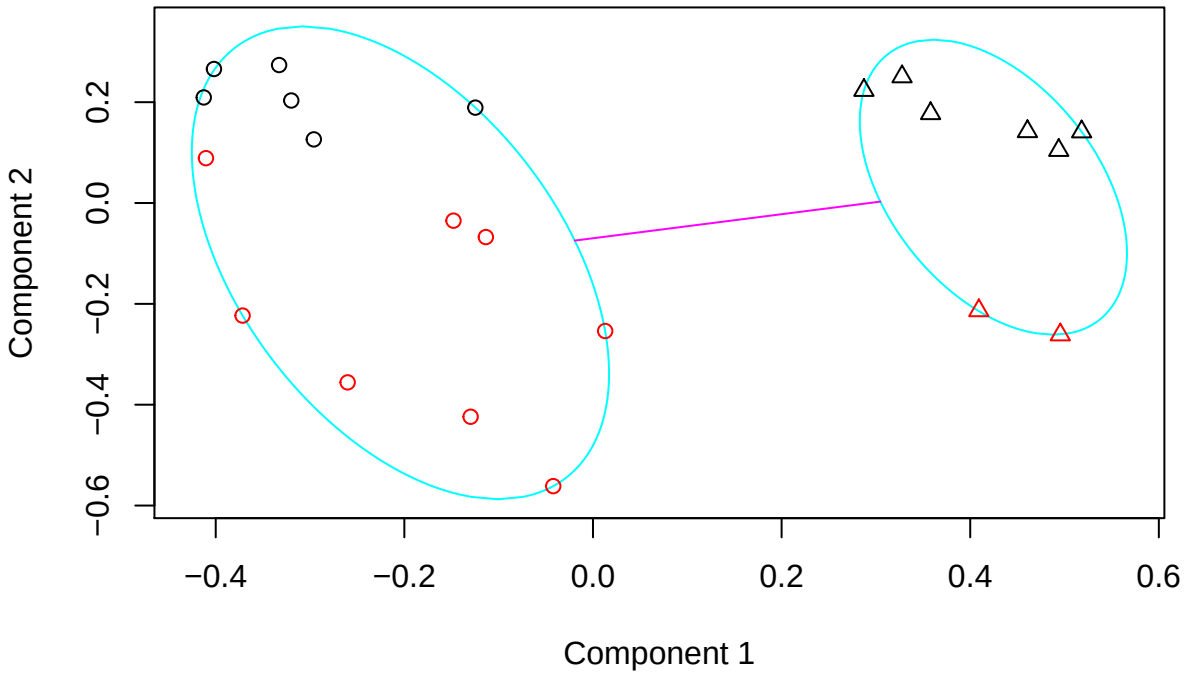
```
# Principal Component Analysis (PCA)
# PCA is a multivariate technique that analyzes a data table
# in which observations are described by several inter-correlated
# quantitative dependent variables. Its goal is to extract the important
# information from the table, to represent it as a set of new orthogonal
# variables called principal components, and to display the pattern of
# similarity of the observations and of the variables as points in maps.
pc_quantile <-prcomp(t(exprs_quantile))
pairs(pc_quantile$x[,1:3],col=eSet_NCI60$status,
      main="Quantile Normalization", legend = TRUE)
```




```
# Loess Normalization

# Heirarchical Clustering
# a graphical representation of a matrix of distances
# a dendrogram, or tree is a plot where the objects are joined
# together in a hierarchical fashion from the closest, that is most
# similar, to the furthest apart, that is the most different
d_loess <- dist2(exprs_loess)
dend_loess <- as.dendrogram(hclust(as.dist(d_loess)))
row.ord_loess <- order.dendrogram(dend_loess)
legend_loess <- list(top = list(fun = dendrogramGrob,
                          args = list(x = dend_loess, side = "top")))
lp_loess <- levelplot(d_loess[row.ord_loess, row.ord_loess],
                      scales = list(x = list(rot = 90, cex=.3), xlab = "", ylab= "",
                                    legend = legend,  legend, y=list(cex=.3)),
main = "Between-Array Distance Plot: Comparing the Expression
      Measures and Sensitivty Status Assigned
      to the Cell Lines in the NCI60 Dataset
      after Loess Normalization")
plot(lp_loess)
```

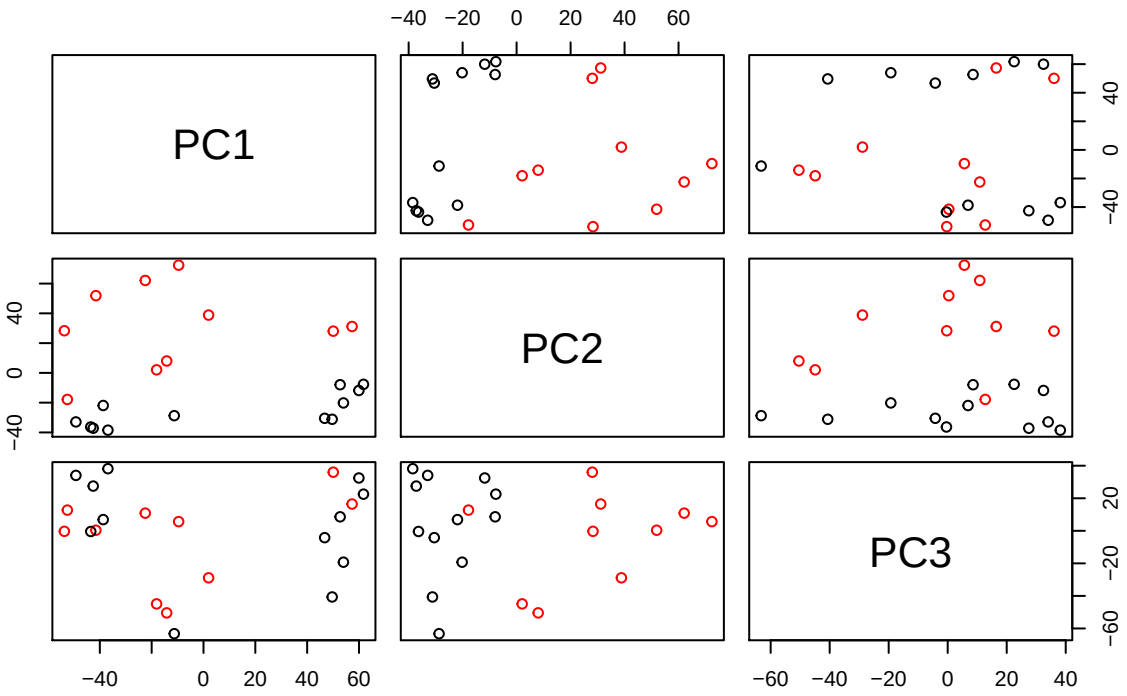



```
# Partitioning Around Medoids (PAM) Clustering
# The algorithm is intended to find a sequence of medoids
# (the distance between two groups as the distance between their centroids
# (center of gravity or vector average) that are centrally located in
# clusters. The goal of the algorithm
# is to minimize the average dissimilarity of objects to their closest
# selected object. Equivalently, we can minimize the sum of the
# dissimilarities between object and their closest selected object.
pl <- cluster::pam(as.dist(d_loess),k=2,diss=TRUE)
cluster::clusplot(pl,col.p=eSet_NCI60$status,
         main="PAM Clustering of NCI60 Dataset after Loess Normalization")
```



These two components explain 29.32 % of the point variability.

```
# Principal Component Analysis (PCA)
# PCA is a multivariate technique that analyzes a data table
# in which observations are described by several inter-correlated
# quantitative dependent variables. Its goal is to extract the important
# information from the table, to represent it as a set of new orthogonal
# variables called principal components, and to display the pattern of
# similarity of the observations and of the variables as points in maps.
pc_loess <-prcomp(t(exprs_loess))
pairs(pc_quantile$x[,1:3],col=eSet_NCI60$status,
      main="Loess Normalization", legend = TRUE)
```



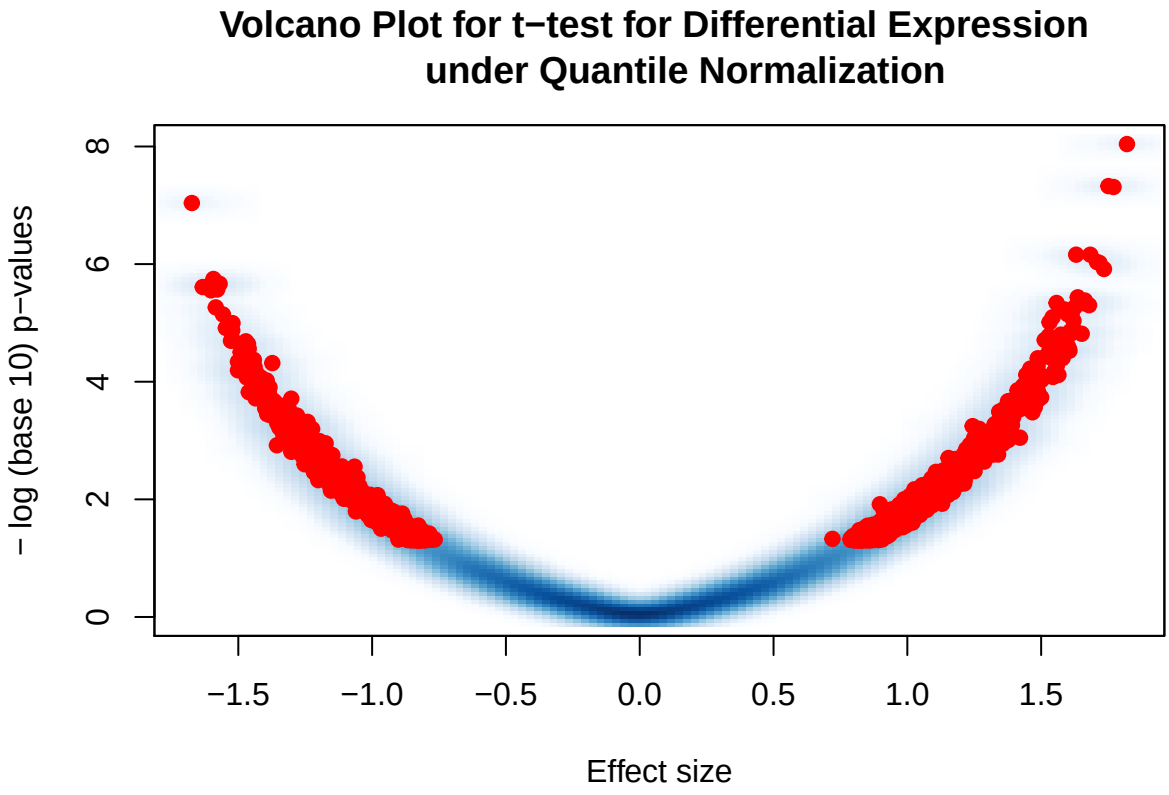
We can see sensitivity status clusering in the principal component plots, especially the second principal component, as well as the partitioning around mediods plots. However, the clusters identified by PAM do not appear to show as much clustering with status groups compared to the PCA plots. These plots are not "perfect" but we do not expect the normalizations to greatly shift our data in different directions. We have a highly complex data structure and there is probably still some variation present between subjects. It would be beneficial if we had a phenotype table with more variables. If we did, we could investigate the role other variables are playing in the variability of our data. Additionally, we have no batch efect information (unless we do and we are not expected to delve into that in this analysis). Regardless, it can be handy to have this information because there are several methods to correct for local unwated variation caused by technical variations in experimental design. Still, overall, we do see more clustering within the sensitivity status groups across both normalizations compared to what we saw before normalization. Thus, the normalization schemes have most likely removed the global batch effects. Perhaps it would be interesting to explore the local batch effect correction methods in the future.

c) Differential expression analysis. Identify genes that are differentially expressed between "Resistant" and "Sensitive" cell lines and compare your list with the signature of Potti et al. (2006) stored in doxorubicinGenes.

In particular, state which test statistics you are using and, in the case of any probabilistic statement, state and justify the underlying assumptions. Provide and comment on numerical and graphical summaries of the results. N.B. Be sure to describe precisely, rigorously, and thoroughly your analyses.

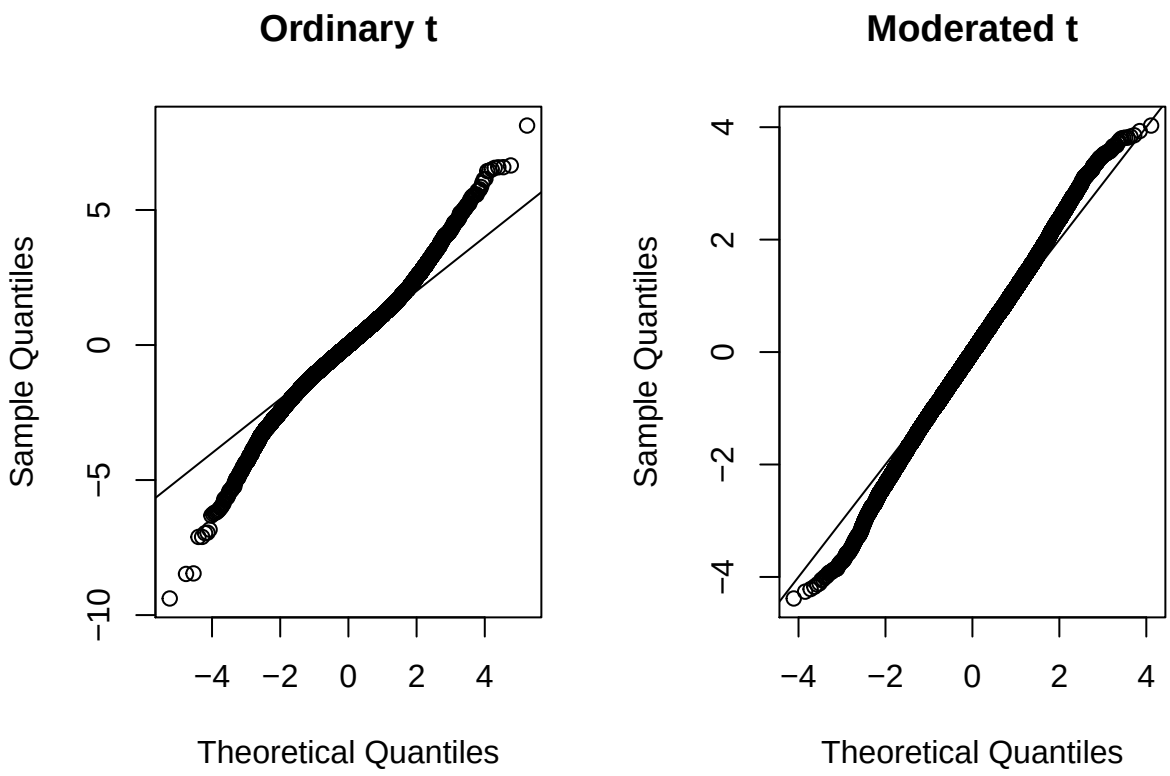
```
# Identify genes that are differentially expressed between
# "Resistant" and "Sensitive" cell lines

# Quantile Normalization

# A common differential expression approach,
# Student's t-test on each row of a matrix using the
# average log-fold changes from two groups
# via genefilter package
```

```
test_quantile <- rowttests(exprs_quantile, eSet_NCI60$status)

# Volcano plot
# By using - log (base 10), the "highly significant"
# features appear at the top of the plot. Using log also
# permits us to better distinguish between small and very
# small p-values. Many features with very small p-values, but
# small effect sizes as we see here, are sometimes indicative
# of problematic data.
ind <- test_quantile$p.value < 0.05
smoothScatter(test_quantile$dm,-log10(test_quantile$p.value),
     xlab="Effect size",ylab="- log (base 10) p-values",
     main = "Volcano Plot for t-test for Differential Expression
     under Quantile Normalization")
points(test_quantile$dm[ind], -log10(test_quantile$p.value)[ind],
       pch = 19, col = "red")
```




```
# getting the top genes for subsequent comparison
test_quantile <- data.frame(gene_ID = rownames(test_quantile),
                            p.value = test_quantile$p.value)
results <- arrange(test_quantile, p.value)
ttop_genes <- results$gene_ID[1:80]

# The empirical Bayes method shrinks the probe-wise sample
# variances towards a common value, augmenting the degrees of
# freedom for the individual variances.
# t-statistics sensitive to variance estimate
# Can improve testing using pooled probe set variances in
```

```r
# an empirical Bayesian approach.
design <- model.matrix(~eSet_NCI60$status)
lm_quantile <- lmFit(exprs_quantile, design)
eb_quantile <- eBayes(lm_quantile)

#  Ordinary t-statistic
ordinary.t <- eb_quantile$coef / eb_quantile$stdev.unscaled / eb_quantile$sigma
#  Q-Q plots of t statistics
#  Points off the line may be differentially expressed
par(mfrow=c(1,2))
qqt(ordinary.t, df=eb_quantile$df.residual, main="Ordinary t")
abline(0,1)
qqt(eb_quantile$t, df=eb_quantile$df.total,main="Moderated t")
abline(0,1)
```




```r
par(mfrow=c(1,1))
# topTable with multiple testing correction
# In general, the adjusted p-values returned by adjust.method="BH"
# remain valid as FDR bounds only when the genes remain sorted by p-value.
ebtop_genes <- topTable(eb_quantile, coef = 2,p.value=1)
dim(ebtop_genes) #only 10 significant genes!
```

```
## [1] 10  6
```

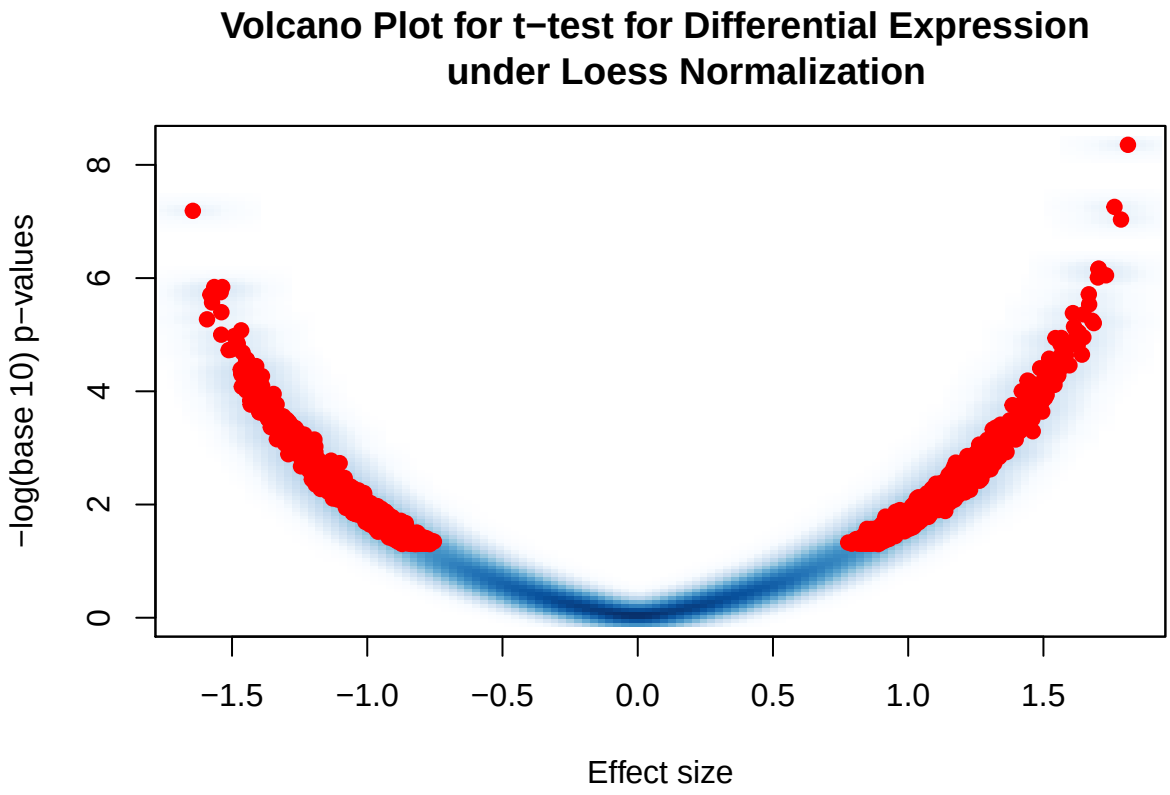
```r
# compare your list with the signature of Potti et al. (2006)
length(which(ttop_genes %in% doxorubicinGenes))
```

```
## [1] 72
```

```r
length(which(rownames(ebtop_genes) %in% doxorubicinGenes))
```

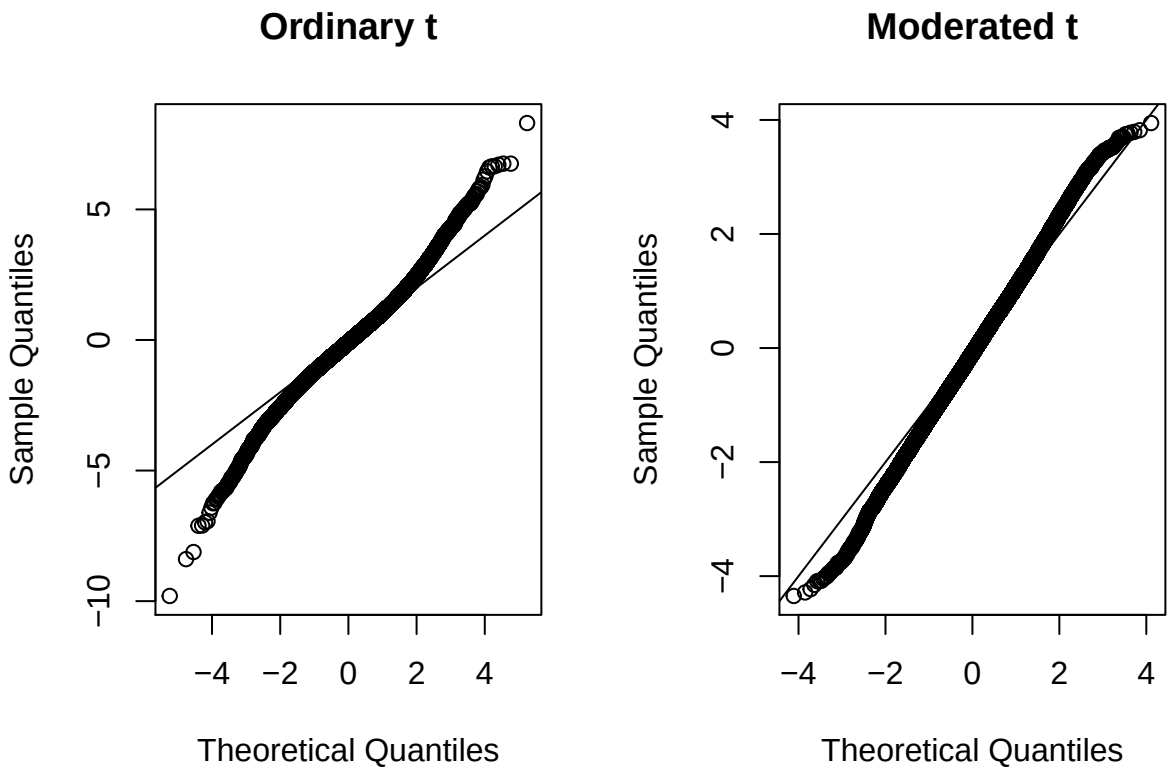
```
## [1] 10
```

```
# Loess Normalization

# Student's t-test on each row of a matrix using the
# average log-fold changes from two groups
# via genefilter
test_loess <- rowttests(exprs_loess, eSet_NCI60$status)

# Volcano plot
# By using - log (base 10), the "highly significant"
# features appear at the top of the plot. Using log also
# permits us to better distinguish between small and very
# small p-values. Many features with very small p-values, but
# small effect sizes as we see here, are sometimes indicative
# of problematic data.
ind <- test_loess$p.value < 0.05
smoothScatter(test_loess$dm,-log10(test_loess$p.value),
     xlab="Effect size", ylab="-log(base 10) p-values",
     main = "Volcano Plot for t-test for Differential Expression
     under Loess Normalization")
points(test_loess$dm[ind], -log10(test_loess$p.value)[ind],
       pch = 19, col = "red")
```




```
# getting the top genes for subsequent comparison
test_loess <- data.frame(gene_ID = rownames(test_loess),
                         p.value = test_loess$p.value)
results <- arrange(test_loess, p.value)
ttop_genes_l <- results$gene_ID[1:80]
```

```
# t-statistics are sensitive to variance estimate
# The empirical Bayes method shrinks the probe-wise sample
# variances towards a common value, augmenting the degrees of
# freedom for the individual variances.
# Can improve testing using pooled probe set variances in
# an empirical Bayesian approach.
design <- model.matrix(~eSet_NCI60$status)
lm_loess <- lmFit(exprs_loess, design)
eb_loess <- eBayes(lm_loess)
#  Ordinary t-statistic
ordinary.t <- eb_loess$coef / eb_loess$stdev.unscaled / eb_loess$sigma
#  Q-Q plots of t statistics
#  Points off the line may be differentially expressed
par(mfrow=c(1,2))
qqt(ordinary.t, df=eb_loess$df.residual, main="Ordinary t")
abline(0,1)
qqt(eb_loess$t, df=eb_loess$df.total,main="Moderated t")
abline(0,1)
```




```
par(mfrow=c(1,1))
# topTable with multiple testing correction
# In general, the adjusted p-values returned by adjust.method="BH"
# remain valid as FDR bounds only when the genes remain sorted by p-value.
ebtop_genes_l <- topTable(eb_loess,coef=2, p.value=1)
dim(ebtop_genes_l) #only 10 significant genes!
```

```
## [1] 10  6
```

```
# compare your list with the signature of Potti et al. (2006)
length(which(ttop_genes_l %in% doxorubicinGenes))
```

```
## [1] 70
```

```
length(which(rownames(ebtop_genes_l) %in% doxorubicinGenes))
```

```
## [1] 10
```

```
rownames(ebtop_genes_l) == rownames(ebtop_genes) #they found five of the same genes!
```

```
##  [1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE
```

Interstingly, for the quantile and loess normalization, the empirical Bayes approach decreased the number of significant genes compared to the probe-wise t-test but all of these genes were found in the Potti et al. (2006) list. Also 5 genes overlapped between the eBayes approach across the normalizations. There was a lot of overlap when doing the standard t-test on each row of the matrix with the Potti genes for both normalizations. To visualize genes that are statistically significant, volcano plots were made for the t-tests. A volcano plot displays log fold changes on the x-axis versus a measure of statistical significance on the y-axis. Here the significance measure can be -log(p-value), which give the posterior log-odds of differential expression. Next, the ordinary t-statistic and moderated t-statistic were plotted using a Q-Q plot for both of the normalizations. In general, Q-Q plots are helpful for examining whether two data sets are generated from the same distribution or not and for determining how well a specified probability distribution fits a sample. Under both normalizations, the moderated t-statistic plot barely strays from the line. This implies that the Gaussian assumption is probably fair to make. The ordinary t-statistic does stray a bit from the normal line and we should be cautious using this ordinary t-statistic for models that assume a normal distribution. Additionally, when we have the moderated t-statistic handy we might as well use that as we know our statistical inference is based on a normal distribution. Next steps of this analysis would incorporate biological validation. With that in mind, we feel most confident about the five genes that overlapped between the Loess and Quantile normalizations under the emprical Bayes framework with adjusted p-values via the Benjamini and Hochberg correction for controlling the FDR, false discovery rate. What is the biological interpretation of the set of differentially expressed genes? Do these genes play an important role in cellular processes? To answer these questions that follow from analysis, we would need to investigate the list of differentially expressed genes further by uncovering the functional information of these genes/biological meaning of them via a reference database where we can map our list of affy_ids to a gene.

Please refer to comments and citations for more information on the statistical methodology and general framework of the approach used in this analysis.

Citations

McCarthy, D. J., and Smyth, G. K. (2009). *Testing significance relative to a fold-change threshold is a TREAT*. Bioinformatics 25, 765-771. http://bioinformatics.oxfordjournals.org/ content/25/6/765

Loennstedt, I., and Speed, T. P. (2002). *Replicated microarray data*. Statistica Sinica 12, 31-46.

Phipson, B, Lee, S, Majewski, IJ, Alexander, WS, and Smyth, GK (2016). *Robust hyperparameter estimation protects against hypervariable genes and improves power to detect differential expression*. Annals of Applied Statistics 10, 946-963. http://projecteuclid.org/euclid.aoas/ 1469199900

Smyth, G. K. (2004). *Linear models and empirical Bayes methods for assessing differential ex- pression in microarray experiments*. Statistical Applications in Genetics and Molecular Biology 3, Article 3. http: //www.statsci.org/smyth/pubs/ebayes.pdf

Processing Affymetrix Expression Data: Bioconductor Course http://master.bioconductor.org/help/ course-materials/2009/SeattleApr09/AffyAtoZ/AffymetrixAtoZSlides.pdf

Dimension reduction for genomics, Jeff Leek: http://jtleek.com/genstats/inst/doc/02_03_dimension-reduction.html

The PAM Clustering Algorithm: https://www.cs.umb.edu/cs738/pam1.pdf

Dr. Michael Greenacre STAT 254 Notes, Stanford: http://84.89.132.1/~michael/stanford/maeb7.pdf

PH252X Series, Biomedicacl Data Science - Volcano Plots: http://genomicsclass.github.io/book/pages/eda_for_highthroughput.html

K. A. Baggerly and K. R. Coombes. Deriving chemosensitivity from cell lines: Forensic bioinformatics and reproducible research in high-throughput biology. *Annals of Applied Statistics*, 3(4):1309–1334, 2009.

K. R. Coombes, J. Wang, and K. A. Baggerly. Microarrays: retracing steps. *Nature Medicine*, 13:1276–1277, 2007.

A. Potti, H. K. Dressman, A. Bild, R. F. Riedel, G. Chan, R. Sayer, J. Cragun, H. Cottrill, M. J. Kelley, R. Petersen, D. Harpole, J. Marks, A. Berchuck, G. S. Ginsburg, P. Febbo, J. Lancaster, and J. R. Nevins. Genomic signatures to guide the use of chemotherapeutics. *Nature Medicine*, 12:1294–1300, 2006.