Breast Cancer Analysis

Note that this paper DOES NOT have an abstract, a discussion, methods, results. IT IS NOT written in an acceptable paper format.

Load the data from the proper directory

```
dir.name <- "/home/philip/workspace/Data Mining/Final Project/Breast Cancer Cells"
#substitute yourOwnPath with the path obtained by getwd()
setwd(dir.name)
```

The affy package is Use the affy pakcage to load the CEL files. The ReadAffy command will load all CEL files of found in the working directory

```
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, xtabs
##
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, as.vector, cbind,
##     colnames, 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, rownames, sapply,
##     setdiff, sort, table, tapply, union, unique, unlist, unsplit
##
## 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")'.

MyData <- ReadAffy()
```

take a look at the data

```
head(MyData)

## Warning in x[seq_len(n)]: The use of abatch[i,] and abatch[i] is
deprecated. Please use abatch[,i] instead.

## Warning: replacing previous import by 'utils::tail' when loading
## 'hgu133acdf'

## Warning: replacing previous import by 'utils::head' when loading
## 'hgu133acdf'

##

## AffyBatch object
## size of arrays=712x712 features (18 kb)
## cdf=HG-U133A (22283 affyids)
## number of samples=4
## number of genes=22283
## annotation=hgu133a
## notes=

str(MyData)

## Formal class 'AffyBatch' [package "affy"] with 10 slots
##   ..@ cdfName          : chr "HG-U133A"
##   ..@ nrow             : Named int 712
##   .. ..- attr(*, "names")= chr "Rows"
##   ..@ ncol             : Named int 712
##   .. ..- attr(*, "names")= chr "Cols"
##   ..@ assayData        :<environment: 0x5580ae0>
##   ..@ phenoData        :Formal class 'AnnotatedDataFrame' [package
"Biobase"] with 4 slots
##   .. .. ..@ varMetadata      :'data.frame':  1 obs. of  1 variable:
##   .. .. .. ..$ labelDescription: chr "arbitrary numbering"
##   .. .. ..@ data             :'data.frame':  4 obs. of  1 variable:
##   .. .. .. ..$ sample: int [1:4] 1 2 3 4
##   .. .. ..@ 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':  506944 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 [1:4] "/home/philip/workspace/Data
Mining/Final Project/Breast Cancer Cells/BT20.CEL"
"/home/philip/workspace/Data Mining/Final Project/Breast Cancer
Cells/BT549.CEL" "/home/philip/workspace/Data Mining/Final
Project/Breast Cancer Cells/HCC2157.CEL" "/home/philip/workspace/Data
Mining/Final Project/Breast Cancer Cells/MDA-MB-231.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 "hgu133a"
##   ..@ protocolData     :Formal class 'AnnotatedDataFrame' [package
"Biobase"] with 4 slots
##   .. .. ..@ varMetadata      :'data.frame':  1 obs. of  1 variable:
##   .. .. .. ..$ labelDescription: chr NA
##   .. .. ..@ data             :'data.frame':  4 obs. of  1 variable:
##   .. .. .. ..$ ScanDate: chr [1:4] "07/03/02 18:21:06" "10/04/02
16:51:26" "08/02/02 17:16:22" "10/04/02 18:47:16"
##   .. .. ..@ 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 2 2
##   .. .. .. ..$ : int [1:3] 2 30 0
##   .. .. .. ..$ : int [1:3] 1 3 0
##   .. .. .. ..$ : int [1:3] 1 2 0
```

```
#figure out what the data is
phenoData(MyData)
```

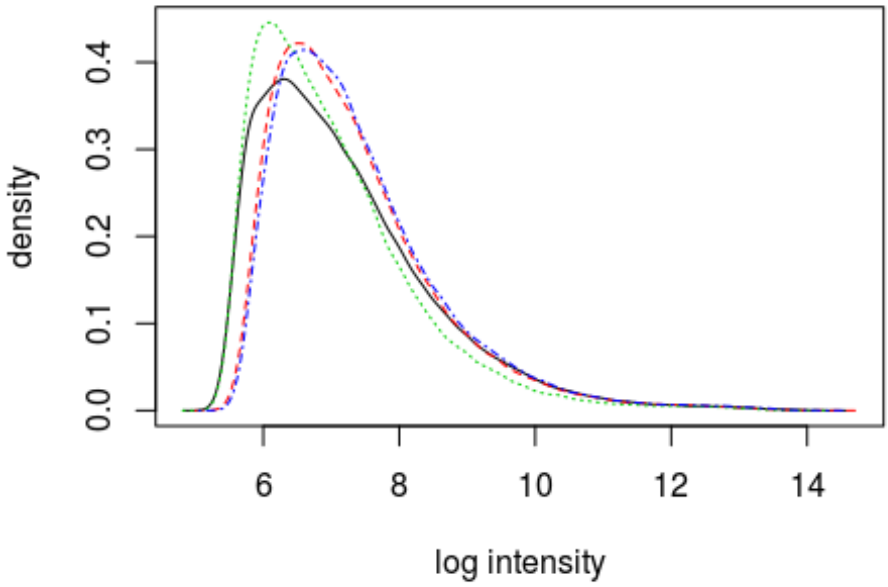
```
## An object of class 'AnnotatedDataFrame'
##   sampleNames: BT20.CEL BT549.CEL HCC2157.CEL MDA-MB-231.CEL
##   varLabels: sample
##   varMetadata: labelDescription
```

```
head(featureNames(MyData))
```

```
## [1] "1007_s_at" "1053_at"   "117_at"    "121_at"    "1255_g_at"
"1294_at"
```

When one visualizes the probe intesities of the values in a density plot, there are four curves for each cell line. It is expected that many of the genes are expressing similiarly between the different cell lines. The majority of the similiarlly expressed genes are probably MM(mismatch values). To correct this systematic bias the data will undergo background corrections and be normalized. After this is completed any remaining differences between cell lines gene expressions should be a result in genetic differences.

```
hist(MyData)
```




```
Normalization of the data should result in the  removal of non-
biological elements of the probe signals.  The two most common
normalization methods are MAS5 and RMA.  Both of these methods deal
with background noise, mismatches, and expression levels differently.
Both methods are widely used by bioinformaticians.  Both methods will
be tested.  [14] The bioconducor package has two options for background
correction methods.
```

```
bgcorrect.methods()
```

```
## [1] "bg.correct" "mas"        "none"       "rma"
```

```
 RMA uses a multi-chip model. It assumes that all chips have the same
background distribution. It does not use mismatch probe intensities
because they typically lead to high variance.  When using RMA smaller
samples will leads to decreased accuracy  but better precision. When
the correction has completed it convert the final normalized values to
log2 values.  [14]
```

MAS5 uses mismatch probe data to create a "robust average". This robust average comes from subtracting mismatch probe values from match probe values. [14]

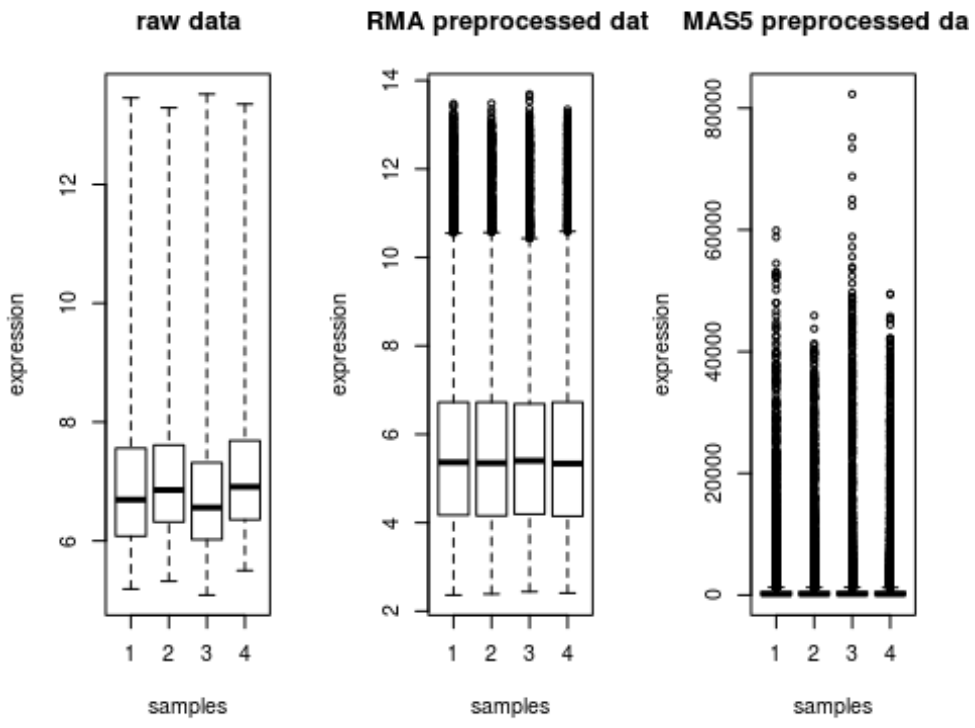
Apply background and normalizaition using methods described above using the rma and mas5 functions available in affy

```
eset1<-rma(MyData)
```

```
## Background correcting
## Normalizing
## Calculating Expression

eset2<-mas5(MyData)

## background correction: mas
## PM/MM correction : mas
## expression values: mas
## background correcting...done.
## 22283 ids to be processed
## |                    |
## |####################|
```

The following box plots visualize the effects of background correction and normalization on the data.

```
par(mfrow=c(1,3)) #Put two plots in one row (2 columns)
boxplot(MyData, col=MyData$samples, xlab="samples", ylab="expression",
names=c(1:4), main="raw data")
boxplot(data.frame(exprs(eset1)), xlab="samples", ylab="expression",
names=c(1:4), main=" RMA preprocessed data")
boxplot(data.frame(exprs(eset2)), xlab="samples", ylab="expression",
names=c(1:4), main=" MAS5 preprocessed data")
```



Now that the data has been normalized it needs to be filtered so the analysis can focus on genes that statistcally significant alevel. The bioconductor package as a genefilter package that can be used to combine several different filters.

```
library(genefilter)
```

```
##
## Attaching package: 'genefilter'
##
## The following object is masked from 'package:base':
##
##     anyNA
```

```
#The first is shapiro wilk test
f1 <- function(x) (shapiro.test(x)$p.value > 0.05)
#The second only keeps genes is the coefficient of variation is smaller
than 0.1
f2 <- function(x) (sd(x)/abs(mean(x))<0.1)
#The thi filter uses a one sample-t-value at a significance of 0.05.
f3 <- function(x) (sqrt(10)* abs(mean(x))/sd(x) > qt(0.975,9))
#combine above 6 functions into one filter
ff <- filterfun(f1,f2,f3)



#Apply the filter on the both datasets
selected1 <- genefilter(eset1[,], ff)
selected2 <- genefilter(eset2[,], ff)
```

After applying the filter the data that was preprocessed using RMA has 524 genes while the data preprocessed using MAS5 has 369 genes.

```
sum(selected1)
```

```
## [1] 17728
```

```
sum(selected2)
```

```
## [1] 386
```

create data sets of expressiong values for both RMA and MAS

```
rma <- eset1[selected1,]
mas<-eset2[selected2,]
annotation(rma)
```

```
## [1] "hgu133a"
```

```
annotation(mas)
```

```
## [1] "hgu133a"
```

***ANOVA ANALYSIS RMA

```
library("limma")
```

```
##
## Attaching package: 'limma'
##
## The following object is masked from 'package:BiocGenerics':
##
##     plotMA
```

```
# tell limma which samples are replicates and which samples belong to
different groups by providing this information in the phenoData slot
ph = rma@phenoData
ph
```

```
## An object of class 'AnnotatedDataFrame'
##   sampleNames: BT20.CEL BT549.CEL HCC2157.CEL MDA-MB-231.CEL
##   varLabels: sample
##   varMetadata: labelDescription
```

```
ph@data[ ,2] = c("BasalA", "BasalB", "BasalA", "BasalB")
colnames(ph@data)[2]="source"
ph@data
```

```
##                sample source
## BT20.CEL            1 BasalA
## BT549.CEL           2 BasalB
## HCC2157.CEL         3 BasalA
## MDA-MB-231.CEL      4 BasalB
```

```
head(rma)
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 1 features, 4 samples
##   element names: exprs
## protocolData
##   sampleNames: BT20.CEL BT549.CEL HCC2157.CEL MDA-MB-231.CEL
##   varLabels: ScanDate
##   varMetadata: labelDescription
## phenoData
##   sampleNames: BT20.CEL BT549.CEL HCC2157.CEL MDA-MB-231.CEL
##   varLabels: sample
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation: hgu133a
```

```
#Tell limma which sample belongs to which group.
groups = ph@data$source
#Convert names of the groups into factors
f = factor(groups, levels = c("BasalA", "BasalB"))

design = model.matrix(~ 0 + f)
colnames(design) = c("BasalA", "BasalB")
```

```r
data.fit = lmFit(rma,design)
#apply Empirical Bayes adjustment.
data.fit.bayes <- eBayes(data.fit)

#print out the top five genes expression values that are basalA
subtypes selected by p-values adjusted by false discovery rate (FDR)
print( topTable(data.fit.bayes, coef=1, number=5, adjust.method="fdr"),
digits=4)
```

```
##             logFC AveExpr     t   P.Value adj.P.Val     B
## 201492_s_at 13.17   13.22 156.4 5.066e-09 1.962e-06 9.721
## 211765_x_at 12.85   12.81 151.9 5.719e-09 1.962e-06 9.695
## 213646_x_at 12.62   12.86 149.0 6.203e-09 1.962e-06 9.677
## 211978_x_at 12.82   12.78 148.6 6.264e-09 1.962e-06 9.674
## 202649_x_at 12.20   12.21 146.8 6.582e-09 1.962e-06 9.663
```

```r
#print out the top five genes expression values that are basalB
subtypes selected by p-values adjusted by false discovery rate (FDR)
print( topTable(data.fit.bayes, coef=2, number=5, adjust.method="fdr"),
digits=4)
```

```
##             logFC AveExpr     t   P.Value adj.P.Val     B
## 201492_s_at 13.27   13.22 157.6 4.905e-09 2.016e-06 9.729
## 213646_x_at 13.10   12.86 154.6 5.310e-09 2.016e-06 9.712
## 211765_x_at 12.77   12.81 151.0 5.858e-09 2.016e-06 9.690
## 211750_x_at 12.96   12.69 148.8 6.226e-09 2.016e-06 9.676
## 211978_x_at 12.75   12.78 147.8 6.404e-09 2.016e-06 9.670
```

```r
#print out the top five genes expression values that of both subtypes
selected by p-values adjusted by false discovery rate (FDR)
print( topTable(data.fit.bayes, number=5, adjust.method="fdr"),
digits=4)
```

```
##             BasalA BasalB AveExpr     F   P.Value adj.P.Val
## 201492_s_at  13.17  13.27   13.22 24639 3.189e-09 1.276e-06
## 213646_x_at  12.62  13.10   12.86 23046 3.667e-09 1.276e-06
## 211765_x_at  12.85  12.77   12.81 22936 3.703e-09 1.276e-06
## 211978_x_at  12.82  12.75   12.78 21967 4.052e-09 1.276e-06
## 202649_x_at  12.20  12.22   12.21 21603 4.196e-09 1.276e-06
```

```r
#design matrix built for the two contrasts
contrast.matrix <- makeContrasts(BasalA-BasalB,levels=design)

#fit linear model with the design matrix
fit1 <- contrasts.fit(data.fit, contrast.matrix)
fit1 <- eBayes(fit1) #apply Empirical Bayes adjustment.

#These are the top ten genes that are differentially expressed bewtween
the two subgroups Basal A and Basal B
#Print the top 10 selected for 2nd coefficient with FDR adjustment,
show only 4 digits.
```

```
K<-
as.data.frame(topTable(fit1,number=100,adjust.method="fdr"),digits=4)
all.K<-
as.data.frame(topTable(fit1,number=17728,adjust.method="fdr"),digits=4)
K[1:10,]
```

```
##                 logFC  AveExpr          t      P.Value adj.P.Val
B
## 201037_at   -1.394303 8.874111 -11.068558 0.0003007376  0.753822 -
3.957436
## 44702_at    -1.286696 7.613845  -9.487648 0.0005603912  0.753822 -
3.968401
## 208804_s_at -1.217093 8.345529  -8.853073 0.0007395289  0.753822 -
3.974408
## 201226_at   -1.188669 8.993646  -8.295452 0.0009583538  0.753822 -
3.980762
## 211954_s_at -1.180172 8.053445  -8.108934 0.0010488542  0.753822 -
3.983158
## 202441_at   -1.031228 7.415576  -7.970526 0.0011228569  0.753822 -
3.985035
## 205142_x_at  1.079766 6.980035   7.969130 0.0011236354  0.753822 -
3.985055
## 201866_s_at -1.013826 6.604674  -7.849627 0.0011928009  0.753822 -
3.986749
## 209067_s_at -1.066381 8.077793  -7.837244 0.0012002630  0.753822 -
3.986928
## 201970_s_at -1.129511 8.525588  -7.793960 0.0012267995  0.753822 -
3.987562
```

***ANOVA ANALYSIS MAS

```
# tell limma which samples are replicates and which samples belong to
different groups by providing this information in the phenoData slot
ph = mas@phenoData
ph
```

```
## An object of class 'AnnotatedDataFrame'
##   sampleNames: BT20.CEL BT549.CEL HCC2157.CEL MDA-MB-231.CEL
##   varLabels: sample
##   varMetadata: labelDescription
```

```
ph@data[ ,2] = c("BasalA", "BasalB", "BasalA", "BasalB")
colnames(ph@data)[2]="source"
ph@data
```

```
##                sample source
## BT20.CEL            1 BasalA
## BT549.CEL           2 BasalB
## HCC2157.CEL         3 BasalA
## MDA-MB-231.CEL      4 BasalB
```

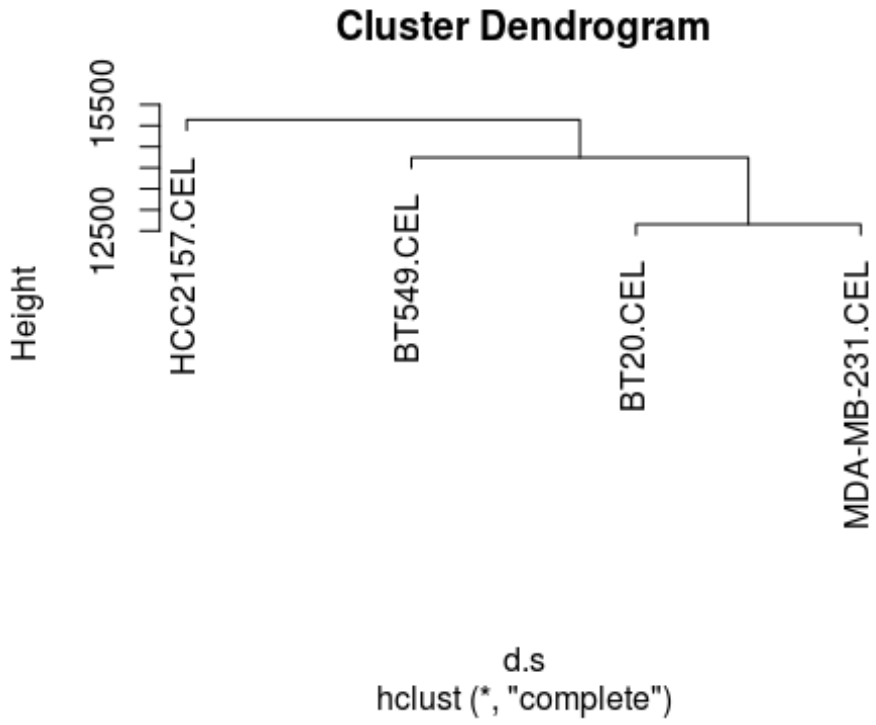
```r
#Tell limma which sample belongs to which group.
groups = ph@data$source
#Convert names of the groups into factors
f = factor(groups, levels = c("BasalA", "BasalB"))

design = model.matrix(~ 0 + f)
colnames(design) = c("BasalA", "BasalB")

data.fit = lmFit(mas,design)
#apply Empirical Bayes adjustment.
data.fit.bayes <- eBayes(data.fit)


#design matrix built for the two contrasts
contrast.matrix <- makeContrasts(BasalA-BasalB,levels=design)

#fit linear model with the design matrix
fit1 <- contrasts.fit(data.fit, contrast.matrix)
fit1 <- eBayes(fit1) #apply Empirical Bayes adjustment.

#These are the top ten genes that are differentially expressed bewtween
the two subgroups Basal A and Basal B
#Print the top 10 selected for 2nd coefficient with FDR adjustment,
show only 4 digits.
M<-
as.data.frame(topTable(fit1,number=100,adjust.method="fdr"),digits=4)
all.M<-
as.data.frame(topTable(fit1,number=length(fit1),adjust.method="fdr"),di
gits=4)
M[1:10,]
```

```
##                   logFC   AveExpr          t     P.Value adj.P.Val
## 209484_s_at -111.96487  713.1384 -15.131119 0.001435410 0.5540683
## 40189_at    1118.02506 6673.8286   8.641761 0.005874512 0.7038553
## 210288_at     29.93984  301.9887   6.633928 0.011286691 0.7038553
## 213264_at     29.37894  203.7251   6.015678 0.014322831 0.7038553
## 215246_at    -33.02260  209.5314  -5.817201 0.015533950 0.7038553
## 201622_at    144.81338 1212.6843   5.710048 0.016246609 0.7038553
## 203777_s_at  -40.22223  341.1195  -5.692868 0.016365007 0.7038553
## 219495_s_at  -23.31941  190.2273  -5.568747 0.017256524 0.7038553
## 207618_s_at -134.66857  925.9067  -5.386255 0.018692173 0.7038553
## 213445_at     48.57290  354.1415   5.279798 0.019605779 0.7038553
##                     B
## 209484_s_at -4.594958
## 40189_at    -4.594963
## 210288_at   -4.594968
## 213264_at   -4.594970
## 215246_at   -4.594971
## 201622_at   -4.594972
## 203777_s_at -4.594972
```

```
## 219495_s_at -4.594973
## 207618_s_at -4.594974
## 213445_at   -4.594974
```

***Hierarchical clustering

```
#Cluster all the data to look for patterns
all<-exprs(mas)
d.s <- dist(t(all))
hc.s <- hclust(d.s, method = "complete")
plot(hc.s)
```




```
groups.s <- cutree(hc.s, k = 2)
table(groups.s, f )
```

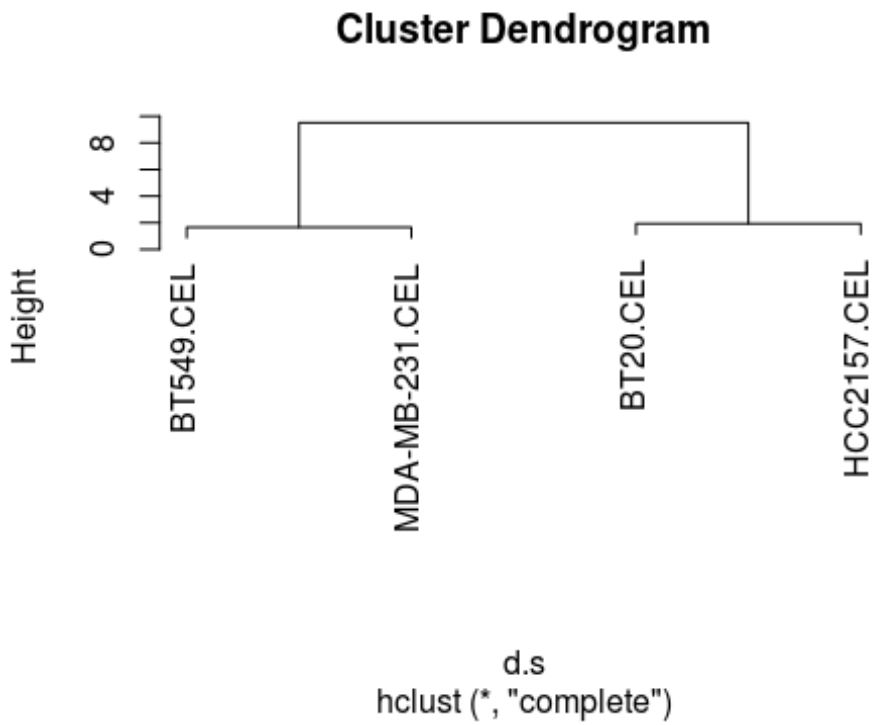
```
##          f
## groups.s BasalA BasalB
##        1      1      2
##        2      1      0
```

Now take the most differentually expressed genes and use them to see if they can be used to properly cluster the cell lines into subtpyes.

*RMA Hierarchical

```
#isolate gene names
genes<-rownames(K)
#subset of data of eset1 of isolated gene names
```

```
genesCl<-eset1[genes,]
#gene expression levels
rma.ex<-exprs(genesCl)
#get the distance between data points
d.s <- dist(t(rma.ex))
hc.s <- hclust(d.s, method = "complete")
plot(hc.s)
```



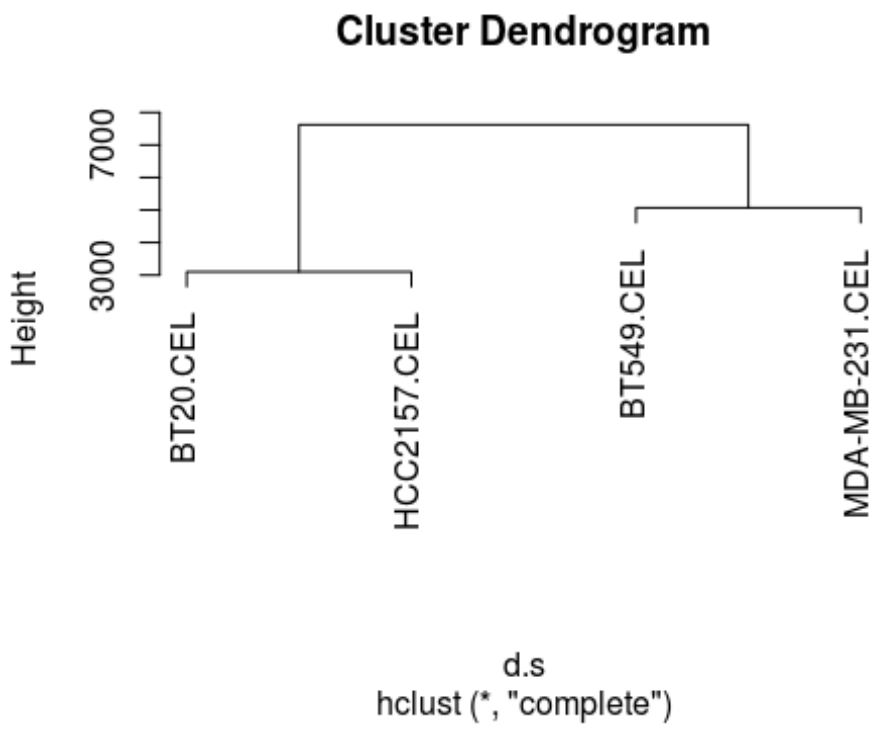

```
#We now split the dendrogram into two clusters (using the function
cutree) and compare the resulting clusters with the true classes.  It
is clear that the clustering of the two different subtypes are much
clearer when using the most differentially expressed genes
groups.s <- cutree(hc.s, k = 2)
table(groups.s, f )
```

```
##          f
## groups.s BasalA BasalB
##        1      2      0
##        2      0      2
```

MAS Hierarchical

```
#isolate gene names
genes<-rownames(M)
#subset of data of eset1 of isolated gene names
genesCl<-eset2[genes,]
#gene expression levels
mas.ex<-exprs(genesCl)
```

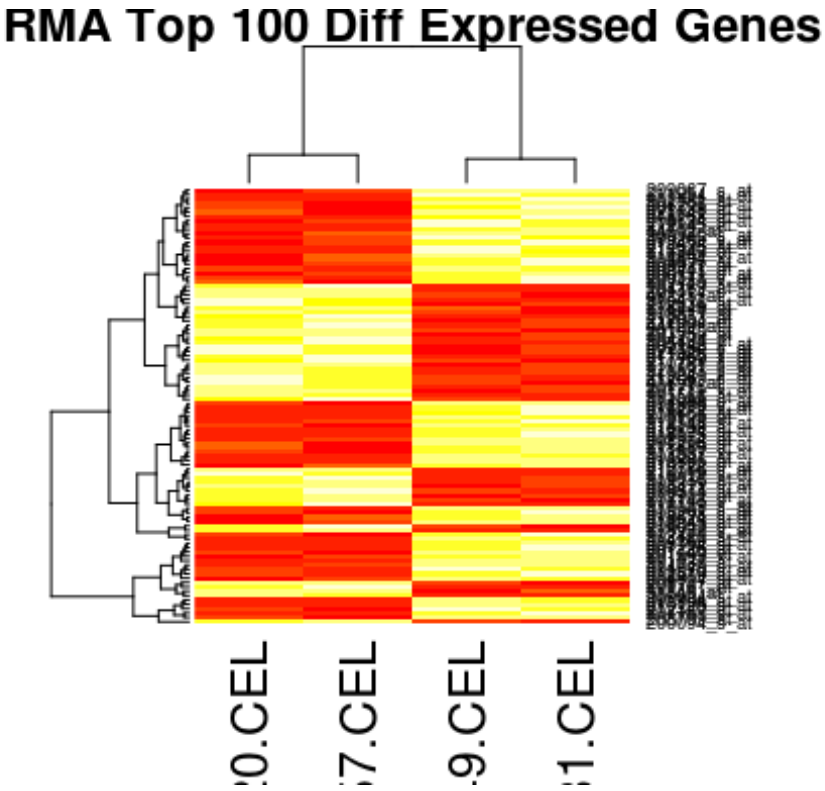
```r
#get the distance between data points
d.s <- dist(t(mas.ex))
hc.s <- hclust(d.s, method = "complete")
plot(hc.s)
```



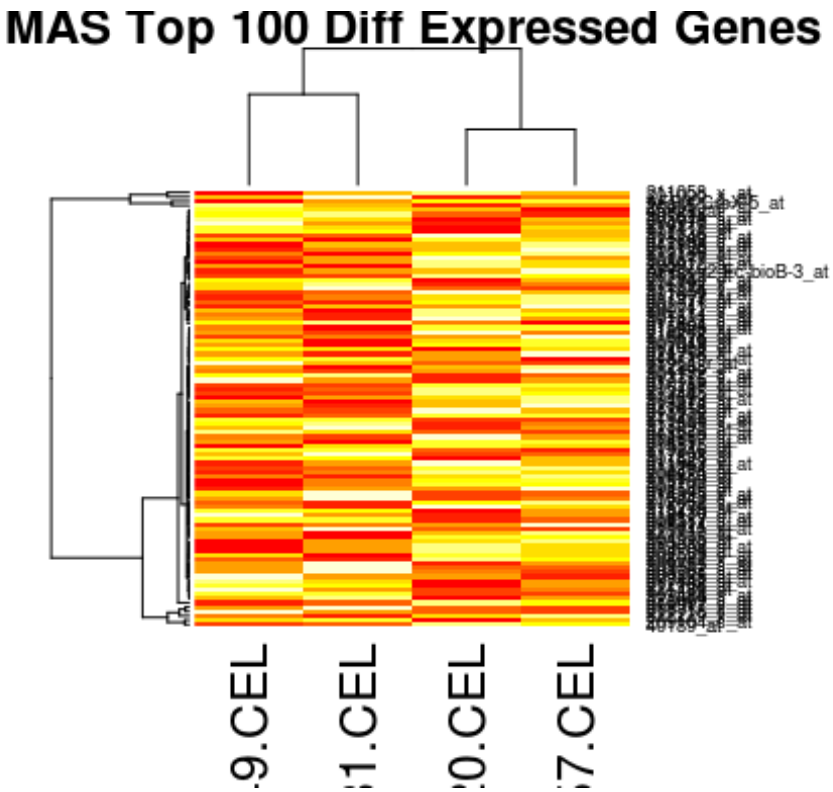

```r
#We now split the dendrogram into two clusters (using the function
cutree) and compare the resulting clusters with the true classes.  It
is clear that the clustering of the two different subtypes are much
clearer when using the most differentially expressed genes
groups.s <- cutree(hc.s, k = 2)
table(groups.s, f )
```

```
##          f
## groups.s BasalA BasalB
##        1      2      0
##        2      0      2
```

```r
#The above pattern is better shown in a heatmap.  When a heatmap is
created with all of the expressiong values it a clear distinction
between subtypes is not evident
#heatmap(all, main = "All Processed Genes")
#When the list of genes is tailored to include the ones with the most
variance and very distinct clusters become evident
heatmap(rma.ex, main = "RMA Top 100 Diff Expressed Genes")
```

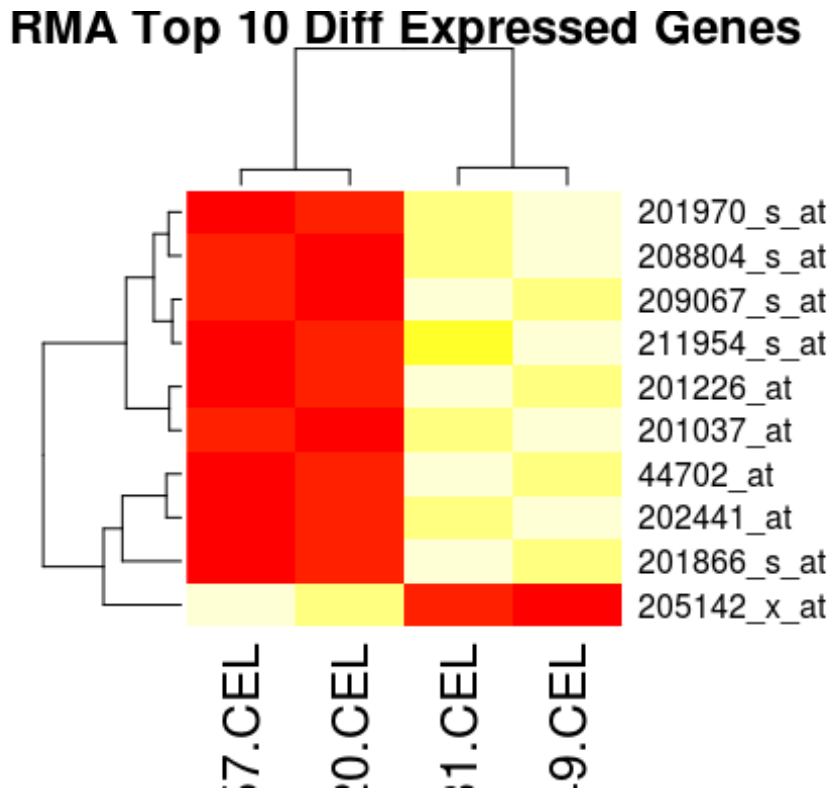



```
heatmap(mas.ex, main = "MAS Top 100 Diff Expressed Genes")
```

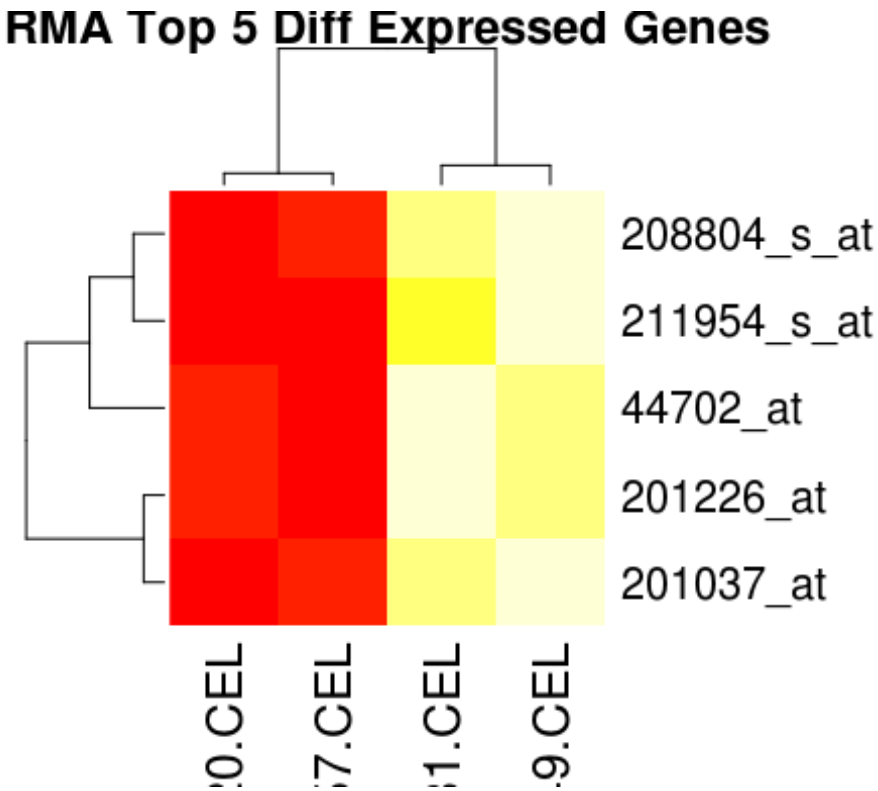


*RMA Lets examine how low we can trim the number of genes to still get a good result.

```r
#Using the top 10 most differentially expressed genes
genes<-rownames(K[1:10,])

genesCl<-eset1[genes,]

rma.ten.ex<-exprs(genesCl)

heatmap(rma.ten.ex, main = "RMA Top 10 Diff Expressed Genes")
```

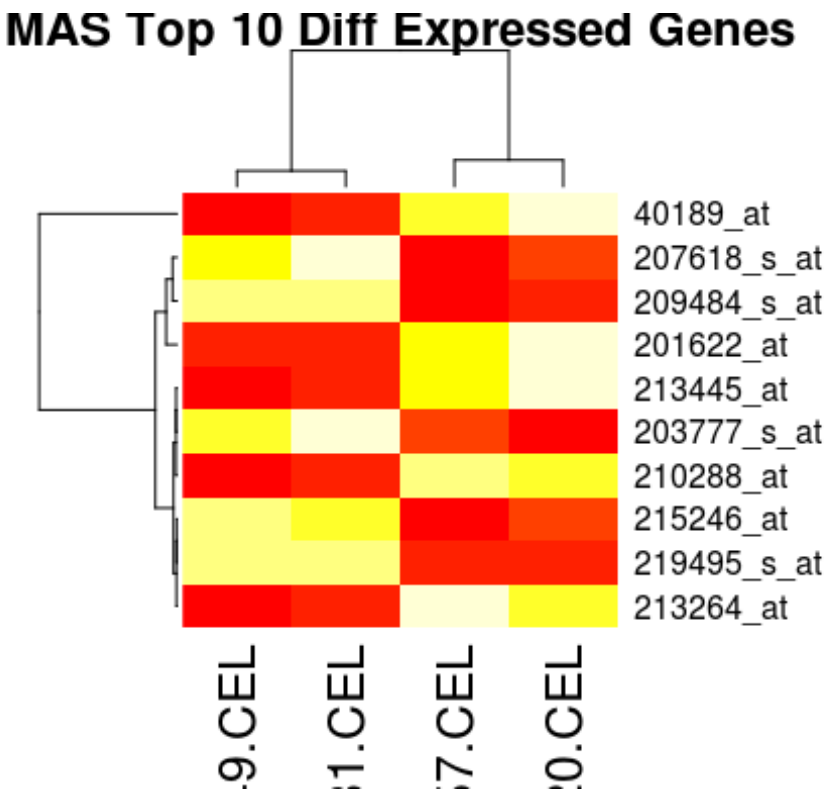



```r
#using the top 5 differentially expressed genes still does a good job
of classifing the subtype
genes<-rownames(K[1:5,])

genesCl<-eset1[genes,]

rma.five.ex<-exprs(genesCl)

heatmap(rma.five.ex,main = "RMA Top 5 Diff Expressed Genes")
```



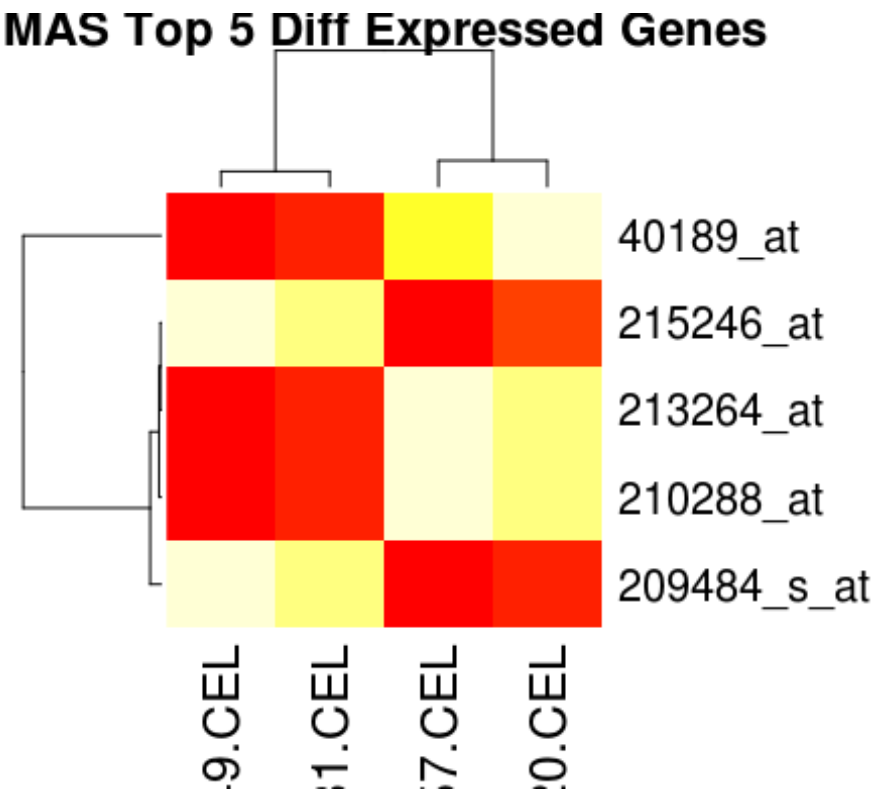
**MAS

```
#lets examine how low we can trim the number of genes to still get a
good result.
#Using the top 10 most differentially expressed genes
genes<-rownames(M[1:10,])

genesCl<-eset2[genes,]

mas.ten.ex<-exprs(genesCl)

heatmap(mas.ten.ex,main = "MAS Top 10 Diff Expressed Genes")
```



Using the top 5 differentially expressed genes still does a good job of classifing the subtype

```
genes<-rownames(M[1:5,])

genesCl<-eset2[genes,]

mas.five.ex<-exprs(genesCl)

heatmap(mas.five.ex, main = "MAS Top 5 Diff Expressed Genes")
```



Compare the most differentially expressed genes from the two normalization algorithms

```
probeset.rma <- as.character(rownames(K)[1:5])
rma.genes<-as.data.frame(probeset.rma)
rma.genes
```

```
##   probeset.rma
## 1    201037_at
## 2     44702_at
## 3  208804_s_at
## 4    201226_at
## 5  211954_s_at
```

```
probeset.mas <- as.character(rownames(M)[1:5])
mas.genes<-as.data.frame(probeset.mas)
```

```
#As you can see despite the normalization technique resulted in
different genes of interest
```

```
top.five<-cbind(rma.genes, mas.genes)
top.five
```

```
##   probeset.rma probeset.mas
## 1    201037_at  209484_s_at
## 2     44702_at     40189_at
```

```
## 3  208804_s_at    210288_at
## 4    201226_at    213264_at
## 5  211954_s_at    215246_at
```

Extract More information about the genes of interest

```
#source("https://bioconductor.org/biocLite.R")
#biocLite("hgu133a.db")
library(hgu133a.db)

## Loading required package: AnnotationDbi
## Loading required package: stats4
## Loading required package: IRanges
## Loading required package: S4Vectors
## Loading required package: org.Hs.eg.db
## Loading required package: DBI

#extract some extra information about the  RMA genes that determine the
subtypes.
select(hgu133a.db,
c("44702_at","201037_at","208804_s_at","211954_s_at","201226_at"),
c("SYMBOL","ENTREZID", "GENENAME"))

## 'select()' returned 1:1 mapping between keys and columns

##       PROBEID SYMBOL ENTREZID
## 1    44702_at  SYDE1    85360
## 2   201037_at   PFKP     5214
## 3 208804_s_at  SRSF6     6431
## 4 211954_s_at   IPO5     3843
## 5   201226_at NDUFB8     4714
##                                                      GENENAME
## 1     synapse defective 1, Rho GTPase, homolog 1 (C. elegans)
## 2                                phosphofructokinase, platelet
## 3                      serine/arginine-rich splicing factor 6
## 4                                                  importin 5
## 5 NADH dehydrogenase (ubiquinone) 1 beta subcomplex, 8, 19kDa

select(hgu133a.db,
c("209484_s_at","40189_at","210288_at","213264_at","215246_at"),
c("SYMBOL","ENTREZID", "GENENAME"))

## 'select()' returned 1:many mapping between keys and columns

##       PROBEID SYMBOL ENTREZID
## 1 209484_s_at   NSL1    25936
## 2    40189_at    SET     6418
## 3    40189_at SETSIP   646817
## 4   210288_at  KLRG1    10219
## 5   213264_at  PCBP2     5094
## 6   215246_at  LARP7    51574
##                                                GENENAME
```

```
## 1              NSL1, MIS12 kinetochore complex component
## 2                              SET nuclear proto-oncogene
## 3                                        SET-like protein
## 4 killer cell lectin-like receptor subfamily G, member 1
## 5                              poly(rC) binding protein 2
## 6           La ribonucleoprotein domain family, member 7
```

Determine what genes overlap from the 100 most differentially expressed genes from the both datasets. The final output shows that there are no shared genes between the top 100 differentially expressed genes of both datasets.

```
#format the probe id's into a format that can be used by 'tm'
probeset.rmaq <- as.character(rownames(K))
rma.genes<-as.data.frame(probeset.rmaq)

probeset.masq <- as.character(rownames(M))
mas.genes<-as.data.frame(probeset.masq)

list<-list(probeset.rmaq, probeset.masq)
list
```

```
## [[1]]
##   [1] "201037_at"   "44702_at"    "208804_s_at" "201226_at"
"211954_s_at"
##   [6] "202441_at"   "205142_x_at" "201866_s_at" "209067_s_at"
"201970_s_at"
##  [11] "201776_s_at" "208764_s_at" "201761_at"   "204091_at"
"214749_s_at"
##  [16] "212866_at"   "212069_s_at" "216272_x_at" "201711_x_at"
"205105_at"
##  [21] "219036_at"   "209513_s_at" "218175_at"   "221893_s_at"
"212145_at"
##  [26] "212624_s_at" "208817_at"   "201740_at"   "203319_s_at"
"204169_at"
##  [31] "203143_s_at" "212053_at"   "212519_at"   "208971_at"
"207157_s_at"
##  [36] "212320_at"   "215471_s_at" "31846_at"    "200721_s_at"
"44120_at"
##  [41] "219961_s_at" "210125_s_at" "209330_s_at" "214088_s_at"
"218448_at"
##  [46] "200094_s_at" "217799_x_at" "209380_s_at" "41858_at"
"213243_at"
##  [51] "208634_s_at" "221641_s_at" "212548_s_at" "200924_s_at"
"218663_at"
##  [56] "201296_s_at" "208706_s_at" "218138_at"   "204404_at"
"201738_at"
##  [61] "218922_s_at" "211537_x_at" "40225_at"    "205298_s_at"
"205566_at"
##  [66] "212635_at"   "211139_s_at" "213346_at"   "219919_s_at"
"208737_at"
```

```
##  [71] "202636_at"   "201945_at"   "212902_at"   "221526_x_at"
"218768_at"
##  [76] "202581_at"   "219241_x_at" "209265_s_at" "206043_s_at"
"201414_s_at"
##  [81] "202974_at"   "208799_at"   "206374_at"   "215411_s_at"
"218886_at"
##  [86] "211465_x_at" "201994_at"   "210761_s_at" "214169_at"
"209445_x_at"
##  [91] "201993_x_at" "212756_s_at" "203856_at"   "203555_at"
"219122_s_at"
##  [96] "212454_x_at" "202422_s_at" "212004_at"   "214894_x_at"
"218624_s_at"
##
## [[2]]
##   [1] "209484_s_at"          "40189_at"             "210288_at"
##   [4] "213264_at"            "215246_at"            "201622_at"
##   [7] "203777_s_at"          "219495_s_at"          "207618_s_at"
##  [10] "213445_at"            "210907_s_at"          "208082_x_at"
##  [13] "221273_s_at"          "201614_s_at"          "201771_at"
##  [16] "211974_x_at"          "215628_x_at"          "213595_s_at"
##  [19] "214395_x_at"          "222229_x_at"          "217973_at"
##  [22] "215100_at"            "218579_s_at"          "205787_x_at"
##  [25] "217804_s_at"          "208587_s_at"          "213136_at"
##  [28] "201918_at"            "218719_s_at"          "38964_r_at"
##  [31] "206792_x_at"          "200670_at"            "202062_s_at"
##  [34] "214137_at"            "207842_s_at"          "204706_at"
##  [37] "208474_at"            "215430_at"            "212708_at"
##  [40] "206880_at"            "215279_at"            "219053_s_at"
##  [43] "203230_at"            "201609_x_at"          "201090_x_at"
##  [46] "AFFX-CreX-5_at"       "218740_s_at"          "213650_at"
##  [49] "204699_s_at"          "208912_s_at"          "217293_at"
##  [52] "217718_s_at"          "208517_x_at"          "AFFX-r2-Ec-
bioB-3_at"
##  [55] "221693_s_at"          "205048_s_at"          "221069_s_at"
##  [58] "208120_x_at"          "212832_s_at"          "208707_at"
##  [61] "203701_s_at"          "200650_s_at"          "203062_s_at"
##  [64] "221612_at"            "216243_s_at"          "201636_at"
##  [67] "201525_at"            "221486_at"            "212388_at"
##  [70] "211863_x_at"          "221758_at"            "201182_s_at"
##  [73] "221627_at"            "209753_s_at"          "211641_x_at"
##  [76] "204549_at"            "209268_at"            "216678_at"
##  [79] "45828_at"             "220710_at"            "209104_s_at"
##  [82] "201421_s_at"          "202041_s_at"          "211058_x_at"
##  [85] "219119_at"            "202553_s_at"          "204680_s_at"
##  [88] "205815_at"            "218179_s_at"          "201877_s_at"
##  [91] "208314_at"            "204995_at"            "217715_x_at"
##  [94] "213314_at"            "214902_x_at"          "200049_at"
##  [97] "200712_s_at"          "204977_at"            "217808_s_at"
## [100] "218262_at"
```

```r
library(tm)
```

```
## Loading required package: NLP
##
## Attaching package: 'NLP'
##
## The following object is masked from 'package:Biobase':
##
##     content
```

```r
myCorpus <- Corpus(VectorSource(list))

inspect(myCorpus[8])
```

```
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 1
##
## [[1]]
## NULL
```

```r
tdm <- TermDocumentMatrix(myCorpus)
tdm
```

```
## <<TermDocumentMatrix (terms: 200, documents: 2)>>
## Non-/sparse entries: 200/200
## Sparsity           : 50%
## Maximal term length: 20
## Weighting          : term frequency (tf)
```

```r
inspect(tdm[,])
```

```
## <<TermDocumentMatrix (terms: 200, documents: 2)>>
## Non-/sparse entries: 200/200
## Sparsity           : 50%
## Maximal term length: 20
## Weighting          : term frequency (tf)
##
##                       Docs
## Terms                  1 2
##   200049_at            0 1
##   200094_s_at          1 0
##   200650_s_at          0 1
##   200670_at            0 1
##   200712_s_at          0 1
##   200721_s_at          1 0
##   200924_s_at          1 0
##   201037_at            1 0
##   201090_x_at          0 1
##   201182_s_at          0 1
##   201226_at            1 0
##   201296_s_at          1 0
```

```
##   201414_s_at          1 0
##   201421_s_at          0 1
##   201525_at            0 1
##   201609_x_at          0 1
##   201614_s_at          0 1
##   201622_at            0 1
##   201636_at            0 1
##   201711_x_at          1 0
##   201738_at            1 0
##   201740_at            1 0
##   201761_at            1 0
##   201771_at            0 1
##   201776_s_at          1 0
##   201866_s_at          1 0
##   201877_s_at          0 1
##   201918_at            0 1
##   201945_at            1 0
##   201970_s_at          1 0
##   201993_x_at          1 0
##   201994_at            1 0
##   202041_s_at          0 1
##   202062_s_at          0 1
##   202422_s_at          1 0
##   202441_at            1 0
##   202553_s_at          0 1
##   202581_at            1 0
##   202636_at            1 0
##   202974_at            1 0
##   203062_s_at          0 1
##   203143_s_at          1 0
##   203230_at            0 1
##   203319_s_at          1 0
##   203555_at            1 0
##   203701_s_at          0 1
##   203777_s_at          0 1
##   203856_at            1 0
##   204091_at            1 0
##   204169_at            1 0
##   204404_at            1 0
##   204549_at            0 1
##   204680_s_at          0 1
##   204699_s_at          0 1
##   204706_at            0 1
##   204977_at            0 1
##   204995_at            0 1
##   205048_s_at          0 1
##   205105_at            1 0
##   205142_x_at          1 0
##   205298_s_at          1 0
##   205566_at            1 0
```

```
##   205787_x_at          0 1
##   205815_at            0 1
##   206043_s_at          1 0
##   206374_at            1 0
##   206792_x_at          0 1
##   206880_at            0 1
##   207157_s_at          1 0
##   207618_s_at          0 1
##   207842_s_at          0 1
##   208082_x_at          0 1
##   208120_x_at          0 1
##   208314_at            0 1
##   208474_at            0 1
##   208517_x_at          0 1
##   208587_s_at          0 1
##   208634_s_at          1 0
##   208706_s_at          1 0
##   208707_at            0 1
##   208737_at            1 0
##   208764_s_at          1 0
##   208799_at            1 0
##   208804_s_at          1 0
##   208817_at            1 0
##   208912_s_at          0 1
##   208971_at            1 0
##   209067_s_at          1 0
##   209104_s_at          0 1
##   209265_s_at          1 0
##   209268_at            0 1
##   209330_s_at          1 0
##   209380_s_at          1 0
##   209445_x_at          1 0
##   209484_s_at          0 1
##   209513_s_at          1 0
##   209753_s_at          0 1
##   210125_s_at          1 0
##   210288_at            0 1
##   210761_s_at          1 0
##   210907_s_at          0 1
##   211058_x_at          0 1
##   211139_s_at          1 0
##   211465_x_at          1 0
##   211537_x_at          1 0
##   211641_x_at          0 1
##   211863_x_at          0 1
##   211954_s_at          1 0
##   211974_x_at          0 1
##   212004_at            1 0
##   212053_at            1 0
##   212069_s_at          1 0
```

```
##   212145_at            1 0
##   212320_at            1 0
##   212388_at            0 1
##   212454_x_at          1 0
##   212519_at            1 0
##   212548_s_at          1 0
##   212624_s_at          1 0
##   212635_at            1 0
##   212708_at            0 1
##   212756_s_at          1 0
##   212832_s_at          0 1
##   212866_at            1 0
##   212902_at            1 0
##   213136_at            0 1
##   213243_at            1 0
##   213264_at            0 1
##   213314_at            0 1
##   213346_at            1 0
##   213445_at            0 1
##   213595_s_at          0 1
##   213650_at            0 1
##   214088_s_at          1 0
##   214137_at            0 1
##   214169_at            1 0
##   214395_x_at          0 1
##   214749_s_at          1 0
##   214894_x_at          1 0
##   214902_x_at          0 1
##   215100_at            0 1
##   215246_at            0 1
##   215279_at            0 1
##   215411_s_at          1 0
##   215430_at            0 1
##   215471_s_at          1 0
##   215628_x_at          0 1
##   216243_s_at          0 1
##   216272_x_at          1 0
##   216678_at            0 1
##   217293_at            0 1
##   217715_x_at          0 1
##   217718_s_at          0 1
##   217799_x_at          1 0
##   217804_s_at          0 1
##   217808_s_at          0 1
##   217973_at            0 1
##   218138_at            1 0
##   218175_at            1 0
##   218179_s_at          0 1
##   218262_at            0 1
##   218448_at            1 0
```

```
##   218579_s_at           0 1
##   218624_s_at           1 0
##   218663_at             1 0
##   218719_s_at           0 1
##   218740_s_at           0 1
##   218768_at             1 0
##   218886_at             1 0
##   218922_s_at           1 0
##   219036_at             1 0
##   219053_s_at           0 1
##   219119_at             0 1
##   219122_s_at           1 0
##   219241_x_at           1 0
##   219495_s_at           0 1
##   219919_s_at           1 0
##   219961_s_at           1 0
##   220710_at             0 1
##   221069_s_at           0 1
##   221273_s_at           0 1
##   221486_at             0 1
##   221526_x_at           1 0
##   221612_at             0 1
##   221627_at             0 1
##   221641_s_at           1 0
##   221693_s_at           0 1
##   221758_at             0 1
##   221893_s_at           1 0
##   222229_x_at           0 1
##   31846_at              1 0
##   38964_r_at            0 1
##   40189_at              0 1
##   40225_at              1 0
##   41858_at              1 0
##   44120_at              1 0
##   44702_at              1 0
##   45828_at              0 1
##   affx-crex-5_at        0 1
##   affx-r2-ec-biob-3_at 0 1
```

```
findFreqTerms(tdm, 2)
```

```
## character(0)
```