Homework4

Carlotta Porcelli - exam_ID 62 - student_ID qbp693

6/13/2017

PART 1

Question 1A: Merge the ChIP peaks that overlap over 1bp or more. How many merged regions are produced compared to how many peaks you started with?

In order to run the *bedtools merge*, the *txnCHIP.bed* file needs to be sorted. This is done by chromosome and then by start position.

```
sort -k1,1 -k2,2n txnChIP.bed > sorted_txnCHIP.bed # presort of .bed file
```

```
wc -l txnChIP.bed
```

The length of the *txnCHIP.bed* file is: 4380444.

Merging the *txnCHIP.bed* file is done not only merging the intervals but also reporting the number of intervals that were integrated into the new file using the **-c 1** (applying *option* on first column) and **-o count** parameters.

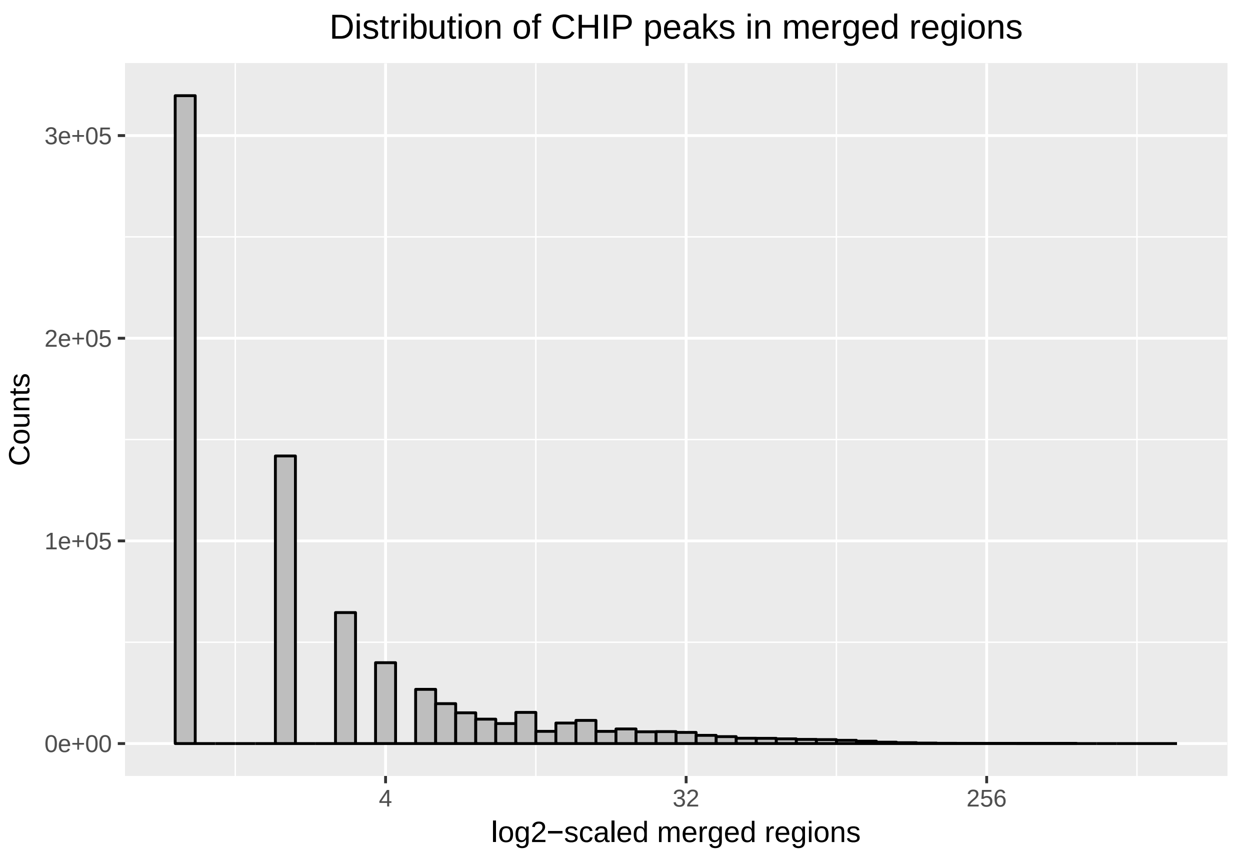
```
bedtools merge -i sorted_txnCHIP.bed -c 1 -o count > count_merged_sorted_txnCHIP.bed
```

```
wc -l count_merged_sorted_txnCHIP.bed
```

The merge tool produced 746610 regions, with a difference of 3633834 regions compared to the initial peaks number.

Question 1B: Plot the distribution of ChIP peaks in each merged region. Briefly comment your plot.

```
library("ggplot2")
CHIP_peaks <- read.table("count_merged_sorted_txnCHIP.bed", header = F)
CHIP_distribution <- ggplot(data=CHIP_peaks, aes(V4)) +
            geom_bar(stat="bin", color='black', fill='grey', binwidth = 0.2) +
            scale_x_continuous(trans='log2') +
            xlab("log2-scaled merged regions") +
            ylab("Counts") +
            ggtitle("Distribution of CHIP peaks in merged regions") +
            theme(plot.title = element_text(hjust = 0.5))
CHIP_distribution
```



The plot shows that more than 3e+05 of the merged regions are composed of only one site. The number of merged regions decrease drastically with the increase of overlapping regions.

Question 1C: Using R, produce a new BED file that contains the 10 merged regions having the highest number of ChIP peaks.

```r
top_10 <- as.data.frame(CHIP_peaks[order(-CHIP_peaks$V4),][0:10,])
write.table(top_10, file='top_10_CHIP_peaks.bed', quote = F, row.names = F, col.names = F)
```

Question 1D: Upload this into the UCSC browser. Look at each merged region and try to interpret it, also taking the peaks it contains into account. What do the merged regions typically overlap? Is there a particular factor that is responsible for the clusters?

Once the top_10 CHIP peaks BED file has been uploaded on the UCSC browser, the track has been mapped to the hg19 assembly. [Screenshots of the 10 merged regions with the highest number of ChIP peaks]

All of the merged regions overlap the RNA polymerase II subunit **POLR2A**. The **POLR2A** forms the largest subunit of RNA polymerase II, enzyme responsible for synthesizing messenger RNA in eukaryotes. In addition, this subunit forms the DNA binding domain of the polymerase, a groove in which the DNA template is transcribed into RNA.Reference:. The transcription factors binding sites are clustered because of the presence of **POLR2A** and its repeatedly presence.

Question 1E: What could we have done to improve the analysis?

To improve the analysis the merge function could be set to combine regions that overlap for more than one basepair. This constraint would produce less overlapping regions.

PART 2

Question 2.1: Provide one line of code which will make symbolic links only to the 6 fastq files.

```
ln -s /home/bohta/HW4/part2/*.fastq /home/qbp693/
```

Question 2.2: Check how well the sequencing run went - Use the 'wc' function to calculate the number of reads in all fastq files and comment on the results.

```
grep '^+$' -c *.fastq # counts the number of lines starting and ending with '+' sign
```

The count of reads for each file is: WT1_R1.fastq:45626717, WT1_R2.fastq:45626717, WT2_R1.fastq:41428670, WT2_R2.fastq:41428670, WT3_R1.fastq:19326183, WT3_R2.fastq:19326183 It is noticeable that there is no difference in number of reads between the strands R1 and R2 of the same library. This is because the two files come from the same cDNA sequence and they have probably been trimmed before removing the orphan reads.

Question 2.3:

A) Report the command for running Kallisto on the WT1 RNA-seq data.

```
nice /home/bohta/bin/kallisto quant --index /home/bohta/HW4/part2/kallistoIndex
    --fr-stranded -t 6 --plaintext --bias -o kallisto_out/ WT1_R1.fastq WT1_R2.fastq
```

B) Report the number of pseudo-aligned reads.

The number of pseudo-aligned reads is 23720001.

C*) Report the estimated average fragment length. Based on this result, what is then the distance between the 3' ends of the two reads in an average read pair? The estimated average fragment length is 178.486.

```
cat WT1_R1.fastq | awk '{if(NR%4==2) print length($1)}' > input_readslength_R1.txt
cat WT1_R2.fastq | awk '{if(NR%4==2) print length($1)}' > input_readslength_R2.txt
```

```
read_len_R1 <- read.table('input_readslength_R1.txt', header=F) # R1 reads length
read_len_R2 <- read.table('input_readslength_R2.txt', header=F) # R2 reads length
r1_mean <- mean(read_len_R1$V1) # average length of R1 reads
r2_mean <- mean(read_len_R2$V1) # average length of R2 reads
avg_fragment_length <- 178.486
```

```
# computes the absolute distance between 3' ends
abs(avg_fragment_length - (r1_mean+r2_mean))
```

The resulting fragment from the computation is a transcript compatible with the mapped reads, this means that the distance between the two 3' ends of a read pair can be computed as the absolute difference between the average_fragment_length and the average length of the reads in R1 and R2. This results in an average distance between the 3' ends of an average read pair of almost 18 bases.

Question 2.4:

A*) Report the command for running Salmon on the WT1 RNA-seq data.

```
nice /home/bohta/bin/salmon quant --index /home/bohta/HW4/part2/salmonIndex -p 6
  --libType A --seqBias --gcBias -1 WT1_R1.fastq -2 WT1_R2.fastq -o salmon_out/
```

B) Report the most likely library type as identify by Salmon.

The automatically detected most likely library is ISF.

C) Report mapping rate.

The mapping rate = 60.9567%.

Question 2.5: Which tool aligned more reads? Comment on the result.

Salmon aligned 27812554 reads, with almost 61% of mapped reads whereas Kallisto mapped almost 52% . Kallisto is based on pseudoalignments to *compatible transcripts*, Salmon is based on lightweight alignments, chains of maximal and super maximal exact matches. Both of the tools mapped above 50% which can be considerated as a good result but surely Salmon performed better.

Question 2.6:

A) Compare the estimated 'effective length' of the isoform 'TCONS_00000020' from Kallisto and Salmon to each other and the reference length.

```
kallisto_quant <- read.table('abundance.tsv', header=T)
salmon_quant <- read.table('quant.sf', header=T)
e_len_kal <- kallisto_quant[kallisto_quant$target_id=='TCONS_00000020',]
e_len_salmon <- salmon_quant[salmon_quant$Name == 'TCONS_00000020',]
e_len_kal
```

```
##         target_id length eff_length est_counts tpm
## 41 TCONS_00000020   4456    4828.72          0   0
```

```
e_len_salmon
```

```
##              Name Length EffectiveLength TPM NumReads
## 41 TCONS_00000020   4456         3842.17   0        0
```

B) What could explain the difference in the effective length? Which estimate do you trust more?

The effective lengths will be affected by the estimated empirical fragment length distribution, the method of calculating effective lengths and whether or not bias correction is used. The most trustworthy is the Salmon computation because it corrects not only the for sequence-specific biases but also for the fragment-level GC biases. Moreover it is unlikely that the effective length is larger than the actual length as Kallisto shows.

PART 3

Question 3.1: Load the data into R. How many isoforms are quantified in the count data?

```
part3_data <- load('part3.Rdata')
nrow(countDF)
```

```
## [1] 5787
```

The number of isoforms quantified in the count data is: 5787.

Question 3.2: Report the R code for how to calculate RPKM values.

```
library_size <- colSums(countDF) # number of reads mapped
transcript_lenghts <- c(annotationDF$length) # vector of lengths of transcripts
# calculation of RPKM values
Rpkm_values <-  as.data.frame(t(t(countDF) * 1000000 / library_size) *
                                as.vector(1000 / transcript_lenghts))
head(Rpkm_values, 2)
```

```
##                K_WT1_Counts K_WT2_Counts K_WT3_Counts S_WT1_Counts
## TCONS_00003947     5.812518     3.087735    7.9140098     5.577944
## TCONS_00003950     2.403414     3.902034    0.8945008     1.156377
##                S_WT2_Counts S_WT3_Counts
## TCONS_00003947     2.249663   7.43761890
## TCONS_00003950     3.452771   0.03223709
```

Question 3.3: Make a one-liner (without the use of ';') that outputs the mean RPKM value of each sample. The restrictions from the previous question no longer apply.

```
mean_values <- colMeans(Rpkm_values) # compute of mean values for each sample
mean_values
```

```
## K_WT1_Counts K_WT2_Counts K_WT3_Counts S_WT1_Counts S_WT2_Counts
##     122.3387     129.5154     121.6085     157.9253     185.6539
## S_WT3_Counts
##     186.2984
```

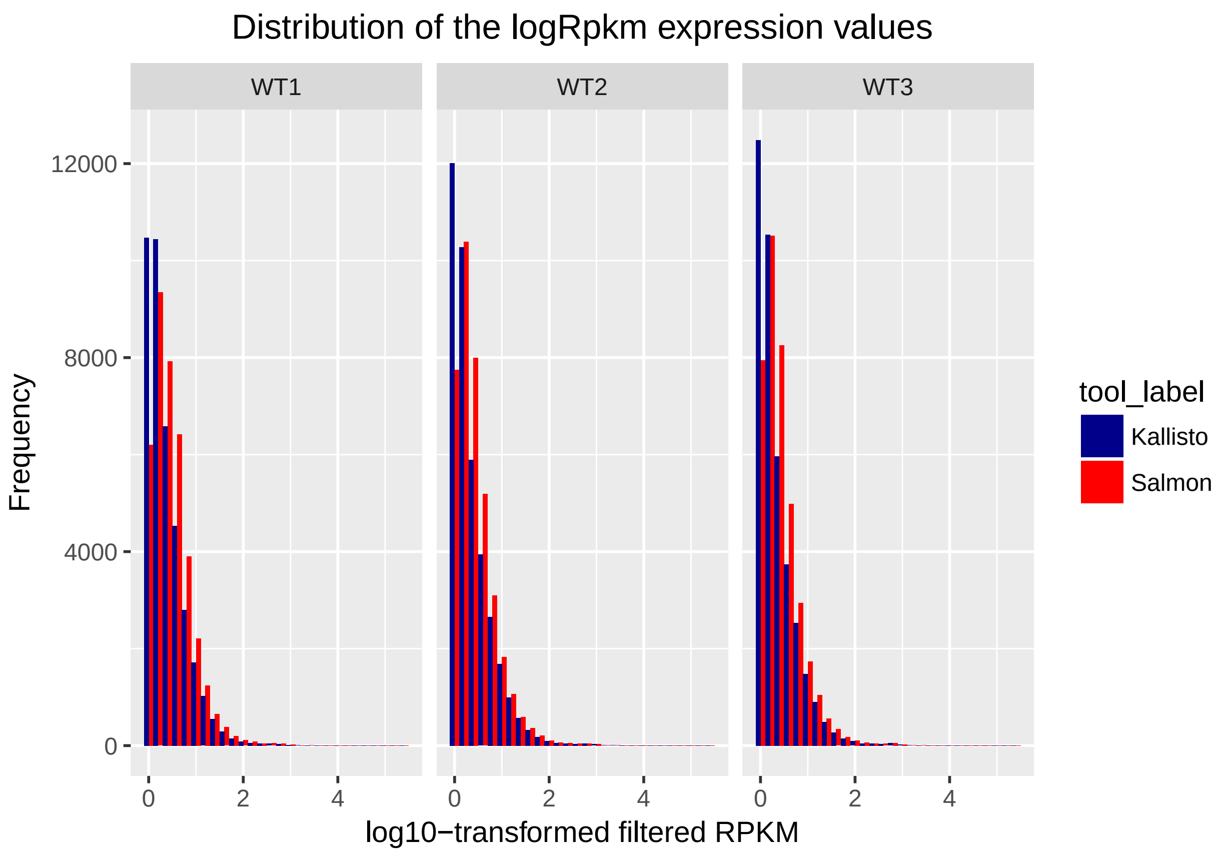
Question 3.4: Make histograms of the distribution of the logRpkm expression values.

```
library(reshape)
library(ggplot2)
logRpkmDF_melt <- melt(logRpkmDF)
```

```
## Using  as id variables
```

```
# adding two columns with labels for tool and for replicates
logRpkmDF_melt$tool_label <- rep(NA, nrow(logRpkmDF_melt))
logRpkmDF_melt$replicate <- rep(NA, nrow(logRpkmDF_melt))
logRpkmDF_melt[grepl("K_", logRpkmDF_melt$variable),][,'tool_label'] <- 'Kallisto'
logRpkmDF_melt[grepl("S_", logRpkmDF_melt$variable),][,'tool_label'] <- 'Salmon'
logRpkmDF_melt[grepl("_WT1_", logRpkmDF_melt$variable),][,'replicate'] <- 'WT1'
logRpkmDF_melt[grepl("_WT2_", logRpkmDF_melt$variable),][,'replicate'] <- 'WT2'
logRpkmDF_melt[grepl("_WT3_", logRpkmDF_melt$variable),][,'replicate'] <- 'WT3'

logRpkm_plot <- ggplot(data=logRpkmDF_melt,
                aes(value, fill=tool_label)) +
                geom_histogram(stat="bin", binwidth = 0.2, position=position_dodge()) +
                xlab("log10-transformed filtered RPKM") +
                ylab("Frequency") +
                ggtitle("Distribution of the logRpkm expression values") +
                theme(plot.title = element_text(hjust = 0.5)) +
                scale_fill_manual(values = c('Kallisto' = "darkblue", 'Salmon' = "red"))+
                facet_wrap(~replicate)
logRpkm_plot
```



Question 3.5: For each tool use logRpkm to calculate all pairwise replicate Pearson correlations of the replicate expression values and report the numbers in a table (one table per tool).

```
logrpmk_table_kallisto <- cor(logRpkmDF[,grepl("K_", names(logRpkmDF))],
                              method = 'pearson')
logrpmk_table_kallisto
```

```
##             K_WT1_RPKM K_WT2_RPKM K_WT3_RPKM
## K_WT1_RPKM  1.0000000  0.9458276  0.9465259
## K_WT2_RPKM  0.9458276  1.0000000  0.9637109
## K_WT3_RPKM  0.9465259  0.9637109  1.0000000
```

```
logrpmk_table_salmon <- cor(logRpkmDF[,grepl("S_", names(logRpkmDF))],
                            method = 'pearson')
logrpmk_table_salmon
```

```
##             S_WT1_RPKM S_WT2_RPKM S_WT3_RPKM
## S_WT1_RPKM  1.0000000  0.9365968  0.9402787
## S_WT2_RPKM  0.9365968  1.0000000  0.9614738
## S_WT3_RPKM  0.9402787  0.9614738  1.0000000
```

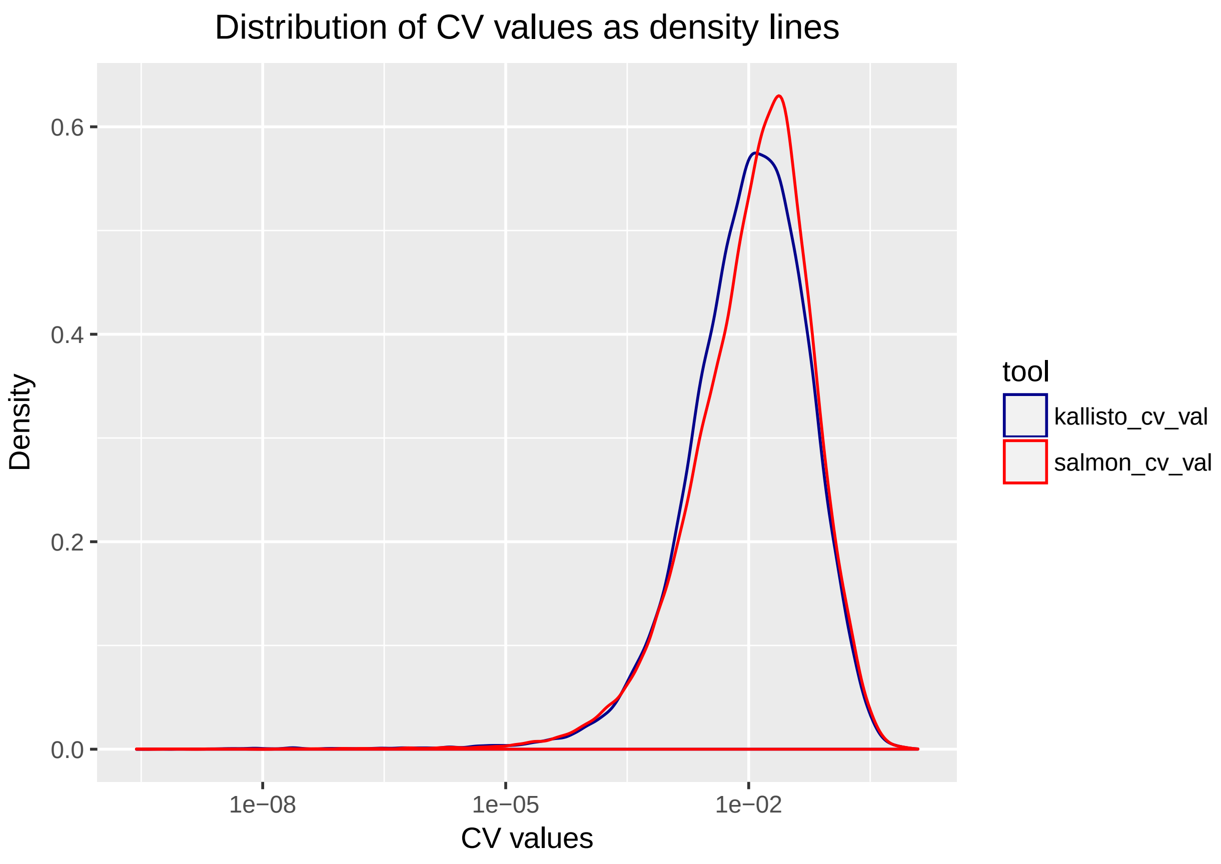
Question 3.6:

A) Construct a function which calculates the CV and use the apply() function to calculate the CV based on logRpkm values for Kallisto and Salmon (separately). 1p

```r
compute_cv <- function(expression_values){
  return (var(expression_values) / mean(expression_values))
}
kallisto_cv_val <- apply(logRpkmDF[,grepl("K_", names(logRpkmDF))], 1, compute_cv)
salmon_cv_val <- apply(logRpkmDF[,grepl("S_", names(logRpkmDF))], 1, compute_cv)
```

B) Plot the distribution of CV values as density lines (in one single plot) using color to indicate the tool and log transform the x-axis (using log10). 1p

```r
library(reshape)
binding_cv <- cbind(kallisto_cv_val, salmon_cv_val)
binding_cv_melted <- melt(binding_cv)
colnames(binding_cv_melted) <- c('transcript', 'tool', 'value')
cv_plot <- ggplot(data=binding_cv_melted,
                aes(value, color=tool)) +
                geom_density(stat="density", position='identity') +
                xlab("CV values") +
                ylab("Density") +
                scale_x_continuous(trans='log10') +
                ggtitle("Distribution of CV values as density lines") +
                theme(plot.title = element_text(hjust = 0.5)) +
                scale_color_manual(values =
                        c('kallisto_cv_val' = "darkblue", 'salmon_cv_val' = "red"))
cv_plot
```



C) Comment on the CV plots using max 75 words. 2p

The plot shows that most of the replicates from Salmon analysis have a cross-replicate variance value greater than 10^-2. From Kallisto analysis, instead, the plot shows that less replicates have a smaller cross-replicate variance value, lower than 10^-2. The cross-replicate variability in the replicates is estimated by the count variance as a function of the mean.

Question 3.7: Based on all the results you have collected here (all of part 2 and all of part 3), discuss in max 100 words which tool would you choose to continue with if you wanted to make a differential expression analysis.

PART 4

```
set.seed(2017)
library(ggplot2)
library(GGally)
```

Question 1: Read both the expression matrix ('ExpressionMatrix.tab') and study design ('StudyDesign.tab') into R. Report the number of samples and the number of genes.

```r
expression_m <- read.table('ExpressionMatrix.tab', header=T) # data import
study_design <- read.table('StudyDesign.tab', header=T) # data import
samples_number <- ncol(expression_m)
samples_number
```

```
## [1] 75
```

```r
gene_number <- nrow(expression_m)
gene_number
```

```
## [1] 10000
```

In the data set there are 75 samples and 10000 genes.

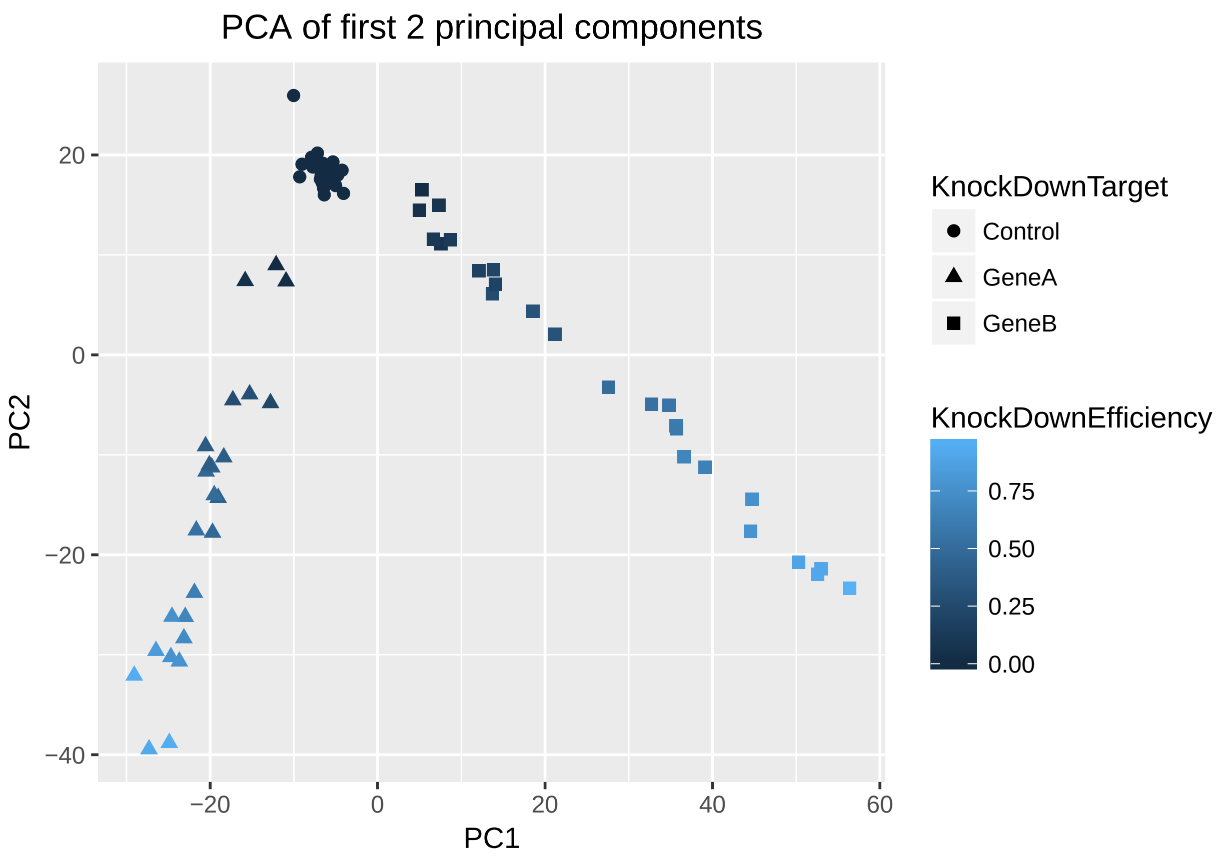
Question 2: Perform PCA on the samples and report the amount of variance contained in the first 5 principle components.

```r
expression_m_transposed <- t(expression_m) # transposition of the matrix
pca_genes <- prcomp(expression_m_transposed, center = T, scale. = T) # pca
```

The expression values have been scaled using center and scale arguments set to TRUE in order to normalize the data. The amount of variance in the first 5 principal components is: PC1: 0.05092, PC2: 0.03281, PC3: 0.02097, PC4: 0.01877, PC5: 0.01525.

Question 3: Make a plot of PC1 vs PC2, where knock down efficiency is indicated by color and knock down target is indicated by shape. Comment on the plot.

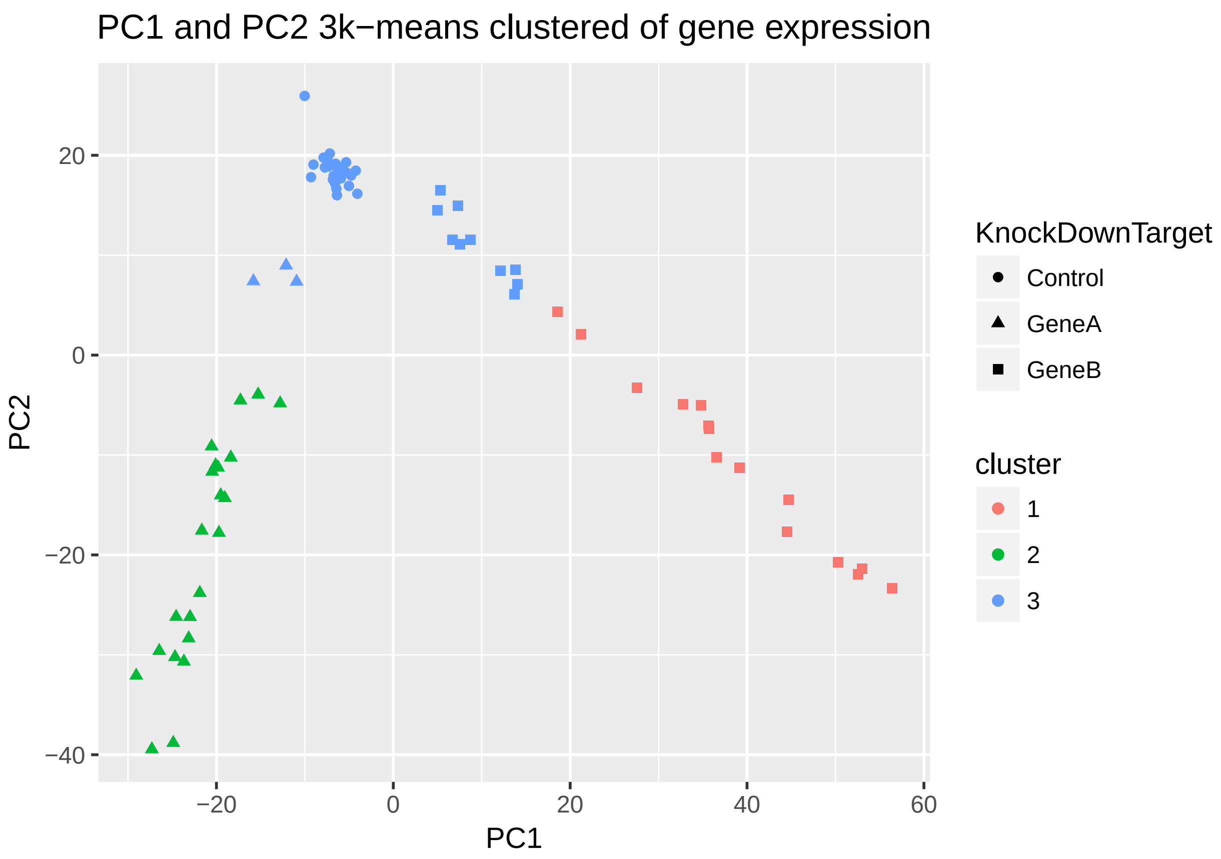
```r
library(ggplot2)
library(GGally)
gene_plot <- data.frame(pca_genes$x, study_design)
ggplot(data=gene_plot, aes(PC1, PC2, color=KnockDownEfficiency, shape = KnockDownTarget)) +
  geom_point(size=2)+
  ggtitle('PCA of first 2 principal components') +
  theme(plot.title = element_text(hjust = 0.5))
```



The plot shows a clearly separated cluster of control genes with knock down efficiency equals to zero. The majority of Gene A samples seem to have a knock down efficiency between 0.50 and 1 on the other end there are three outliers closer to the control cluster with efficiency closer to zero. The samples from the Gene B are well clustered on the right side of the control cluster. They span the across the range of efficiency values but still a small amount of samples has knockdown efficiency closer to zero.

Question 4: Perform a K-means clustering of the data, using k=3 and 10 random starting points. Visualize the clustering by making a plot of PC1 vs PC2, where the clusters are indicated by color. Briefly comment on how the clustering corresponds to the known knockdown targets.

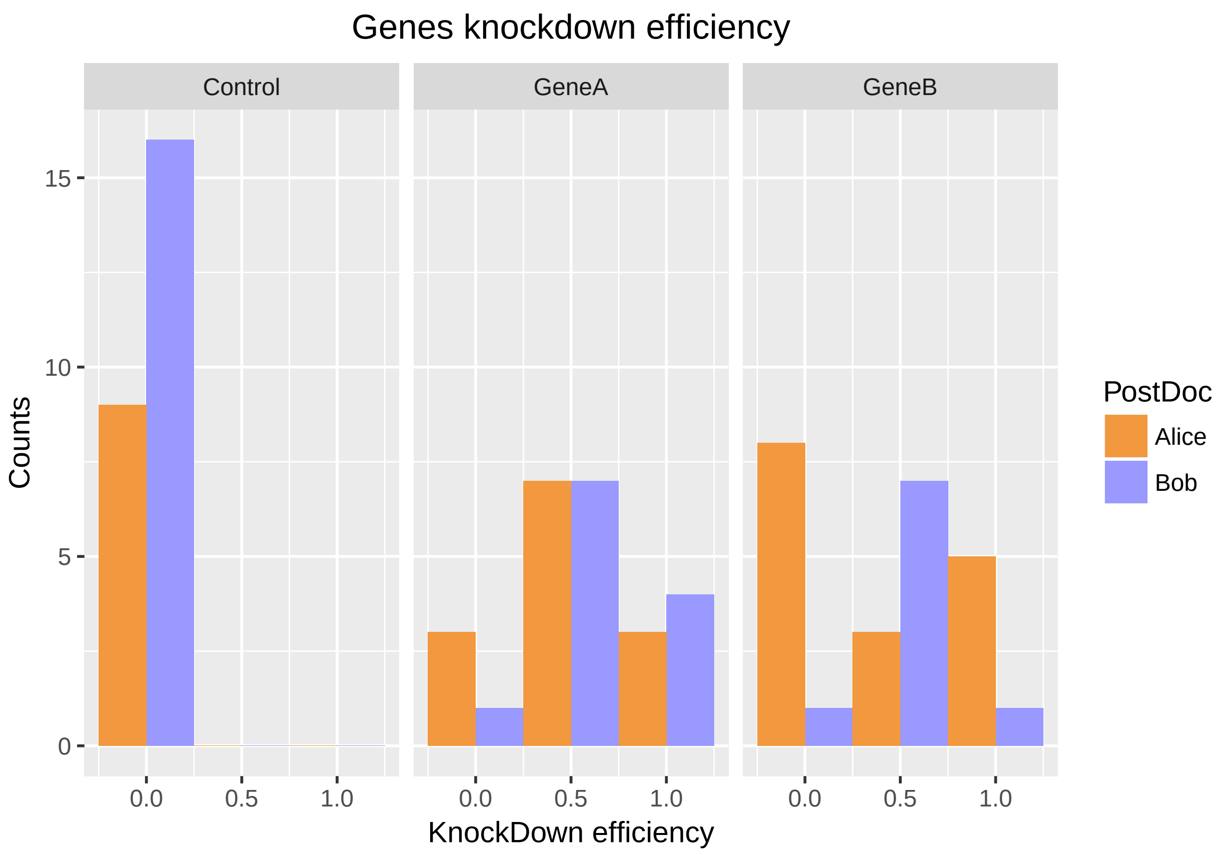
```r
# 3-means with 10 starting points
k3_means <- kmeans(expression_m_transposed, centers=3, nstart = 10)
gene_plot$cluster <- factor(k3_means$cluster) # adding clusters to data frame
# plotting results
k_means_plot <- ggplot(data=gene_plot, aes(x=PC1, y=PC2, color=cluster,
                                          shape = KnockDownTarget)) +
  geom_point(size=1.5) +
  ggtitle('PC1 and PC2 3k-means clustered of gene expression') +
  theme(plot.title = element_text(hjust = 0.5))
k_means_plot
```



The plot shows that the middle cluster contains samples with knock the down efficiency closer to zero. This cluster should contain only control samples so it is evident that something in the experiment didn't work as expected. The cluster on the right side groups all the samples of Gene B and the other cluster on the left contains all the samples from Gene A with higher knock down efficiencies.

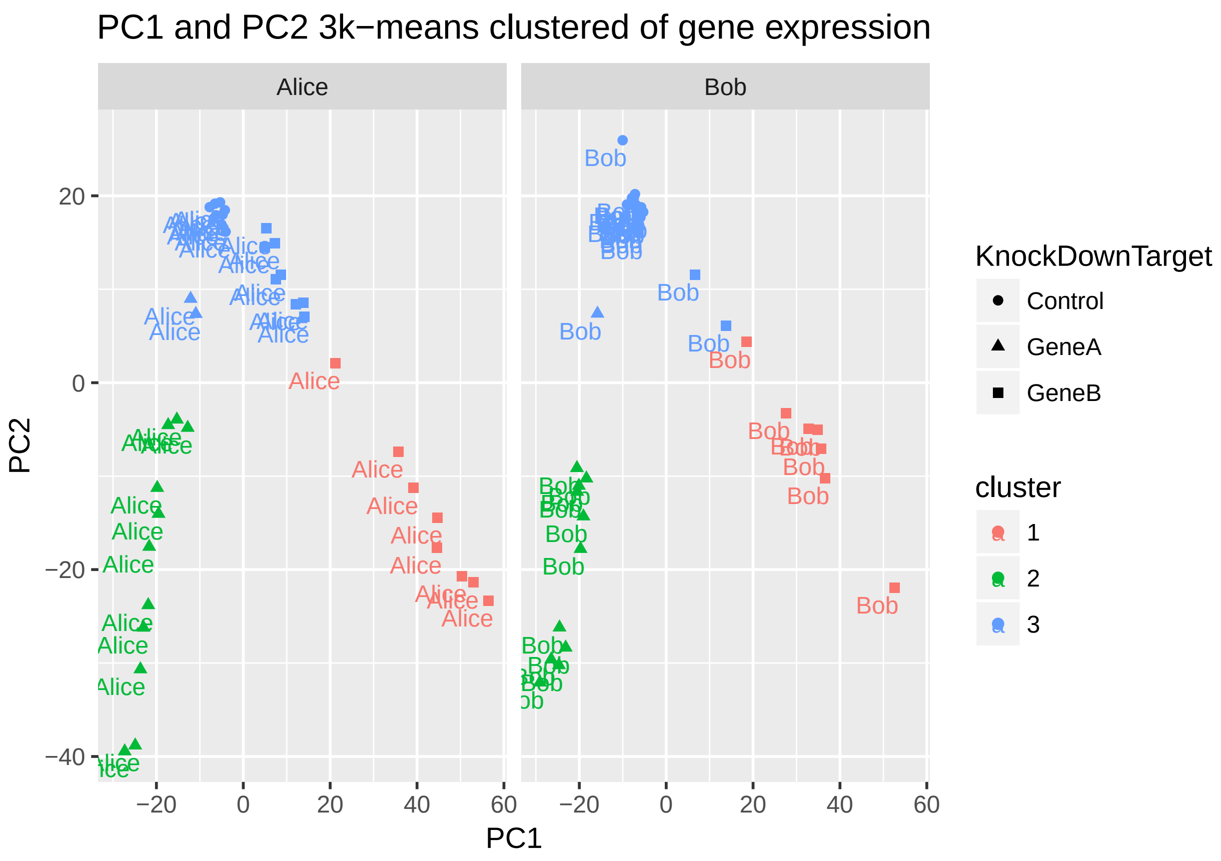
Question 5: Using a maximum number of 3 plots, investigate whether there is any truth to the postdoc allegations: Can you see a difference in samples prepared by Alice and Bob? Is there any indication Bob has ruined a sample? Discuss you results using a maximum of 75 words.

```r
library(ggrepel)
KD_effiency <- ggplot(data=study_design,
            aes(KnockDownEfficiency, fill=PostDoc)) +
            geom_bar(stat="bin", binwidth = 0.5, position=position_dodge()) +
            scale_fill_manual(values = c("Bob" = "#9999ff", "Alice" = "#f2983e"))+
            xlab("KnockDown efficiency") +
            ylab("Counts") +
            ggtitle("Genes knockdown efficiency") +
            theme(plot.title = element_text(hjust = 0.5))+
            facet_wrap(~KnockDownTarget)
KD_effiency
```



All the Control samples from both PostDocs have KnockDownEfficiency equals to zero, as expected. For Gene A, Bob got fewer samples with 0 efficiency than Alice. For Gene B Alice appears to have more samples with efficiency value equals to zero than Bob.

```
k_means_plot + geom_text(aes(label=PostDoc), size=3, hjust=0.9, vjust=1.5) + facet_wrap(~PostDoc)
```



From the plot with PostDoc labels it is clear that Bob might have affected one of the Control sample as it is significantly further away from the samples in the cluster.

Question 6: Based on all you observations in Question 1-5, discuss whether the experiment is still useful, or whether the postdocs have ruined it. Use a maximum of 100 words. 3p

From all the observations it looks like Alice has a good reason for saying that Bob has sneezed in one of the sample because of the outliers shown from the PCA plots. Moreover, it has to be noted that a considerate number of samples from Alice are clustered wrongly in the Control cluster because of their KnockDownEfficiency value. That said, the eperiment could still be useful to some extent removing some of the data that looks compromised. On the other end it is true that if the experiment is need for a more accurate analysis it should probably be performed again.

PART 5

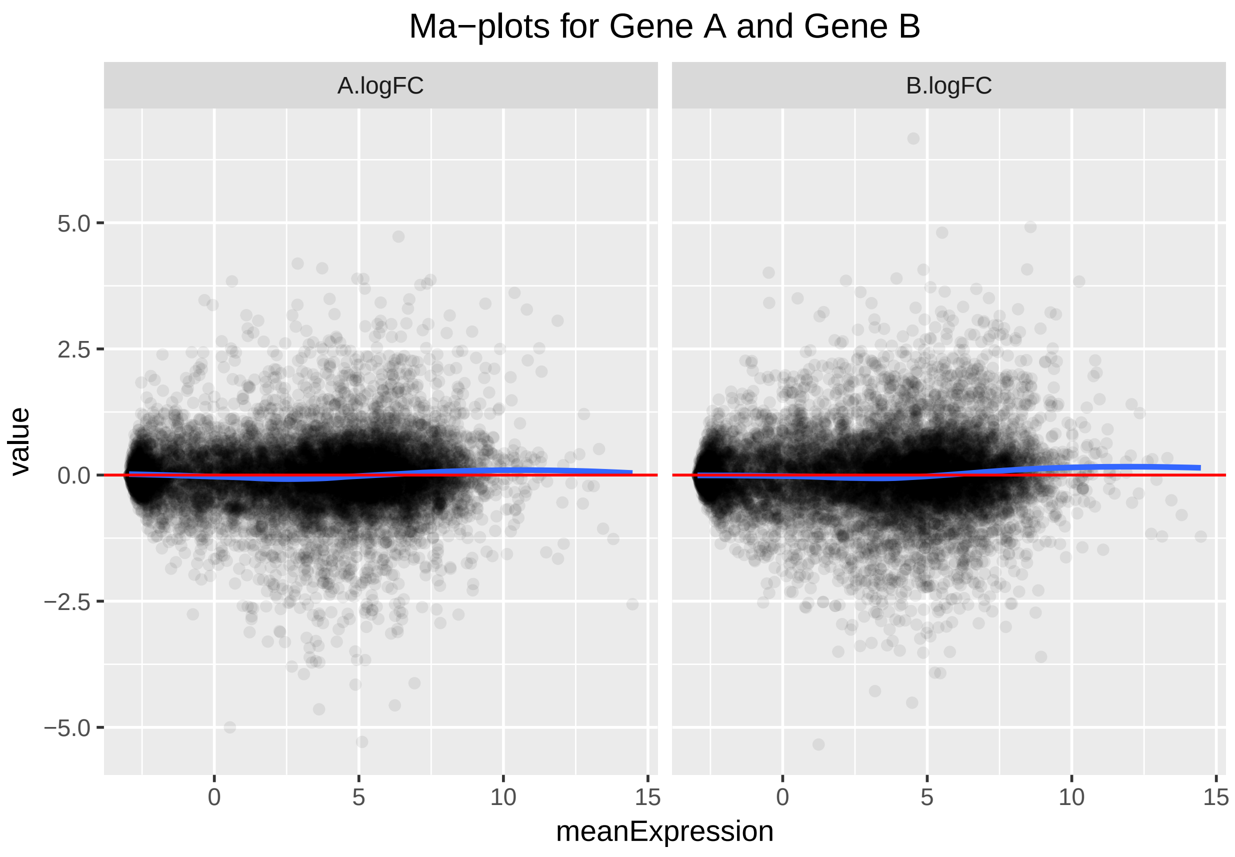
Question 1: Read the DE analysis ('DifferentialExpression.tab) results into R, and show the first few lines of the file.

```
differential_expression <- read.table('DifferentialExpression.tab', header=T)
head(differential_expression, 3)
```

```
##    gene meanExpression    A.logFC  A.pvalue     A.FDR     B.logFC
## 1 Gene1     -2.5530584  0.6113569 0.0624194 0.3605973 -0.06796082
## 2 Gene2     10.8615216  0.1950991 0.7051621 0.9458926  2.02880013
## 3 Gene3      0.4656058 -0.4424548 0.3237347 0.7569917 -0.49773125
##       B.pvalue      B.FDR
## 1 8.250010e-01 0.94580106
## 2 7.077611e-05 0.00132045
## 3 2.411661e-01 0.59217398
```

Question 2: As part of a single plot, produce two MA-plots (one for each knock-down). To mitigate overplotting, points on the plot should be transparent

```r
library(ggplot2)
library(GGally)
library(gridExtra)
library(reshape)
melt_de <- melt(differential_expression, measure.vars =c('A.logFC', 'B.logFC'))
ma_plots <- ggplot(data = melt_de, aes(x = meanExpression, y = value)) +
  geom_point(size=1.5, alpha=0.07) +
  geom_smooth(method = 'loess', se=FALSE) +
  geom_hline(yintercept = 0, col='red') +
  ggtitle("Ma-plots for Gene A and Gene B") +
  theme(plot.title = element_text(hjust = 0.5)) +
  facet_wrap(~variable)
ma_plots
```



Question 3: Explain how an MA-plot can be used to investigate whether expression data has been properly normalized (max 75 words).

Question 4: Report the number of upregulated and downregulated genes that are significantly DE after correction for multiple testing (at alpha=0.05). Which knockdown target has the highest total number of DE genes?

```r
de_genes_a <- differential_expression[which(differential_expression$A.FDR<0.05),]
down_reg_a <- nrow(de_genes_a[de_genes_a$A.logFC<0,])
down_reg_a
```

```
## [1] 374
```

```r
up_reg_a <- nrow(de_genes_a[de_genes_a$A.logFC>0,])
up_reg_a
```

```
## [1] 380
```

```r
de_genes_b <- differential_expression[which(differential_expression$B.FDR<0.05),]
down_reg_b <- nrow(de_genes_b[de_genes_b$B.logFC<0,])
down_reg_b
```

```
## [1] 645
```

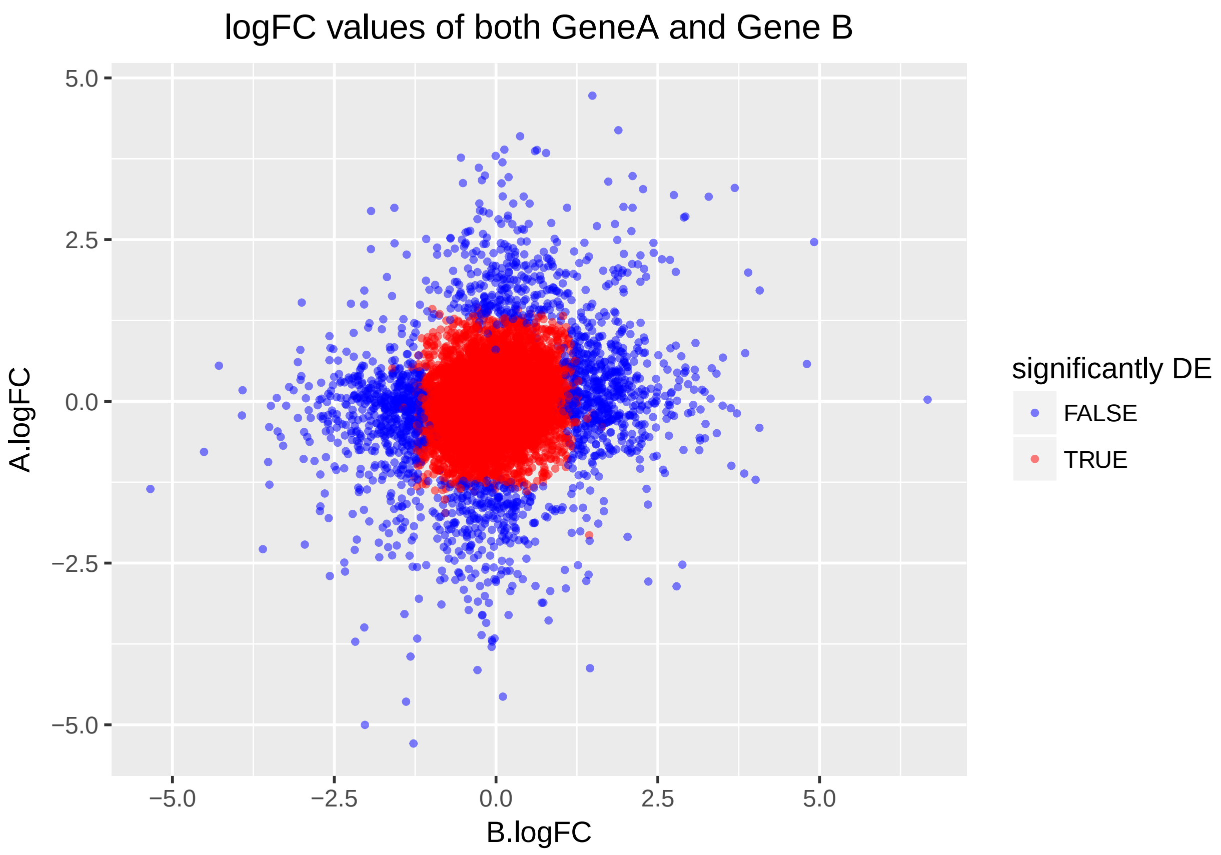
```r
up_reg_b <- nrow(de_genes_b[de_genes_b$B.logFC>0,])
up_reg_b
```

```
## [1] 659
```

For GeneA the number of downregulated genes is 374 and the number of upregulated genes is 380. For GeneB the number of downregulated genes is 645 and the number of upregulated genes is 659, this is the target with the highest number of DE genes: 1304 genes.

Question 5: Plot the two sets of logFCs (log2 fold changes) against each other. Color points based on whether they are significantly DE after multiple testing correction (at alpha=0.05) in either one or both of the knockdowns.

```r
log_plot <- ggplot(data = differential_expression) +
  geom_point(aes(B.logFC, A.logFC, colour = (A.FDR>0.05 & B.FDR>0.05)),
             size=0.8, alpha=0.5) +
  scale_colour_manual(name = 'significantly DE', values = setNames(c('red','blue'),
                                                                   c(T, F))) +
  ggtitle("logFC values of both GeneA and Gene B") +
  theme(plot.title = element_text(hjust = 0.5))
log_plot
```



Question 6: Calculate the Pearson correlation between the logFCs of the two knockdowns, and test whether this correlation is significant.

```r
p_correlation <- cor(differential_expression$A.logFC, differential_expression$B.logFC,
                     method="pearson")
p_correlation
```

```
## [1] 0.1829156
```

The correlation coefficient is 0.1829156 denoting a very weak positive correlation between the two knockdowns.

Question 7: Caroline asks you to construct a 2-by-2 contingency table showing whether genes are significantly DE (after correcting for multiple testing, alpha=0.05) in the two knockdowns. Perform a Fisher???s Exact test for any association between the DE genes in the two knockdowns.

```r
de_table <- table(differential_expression$A.FDR<0.05, differential_expression$B.FDR<0.05)
fisher.test(de_table)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  de_table
## p-value = 2.116e-11
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  1.604151 2.340773
## sample estimates:
## odds ratio
##   1.941789
```

alternative: there is correlation between samples low corrlation - p value is low -> ## Question 8: Based on Question 4-7, do you think the geneA and geneB transcription factors regulate the same genes? Use a maximum of 100 words. the differential expression is different - no