Part 3 codes

Annotating and combining the DE analysis files helps visualise data in greater resolution. The FPKMs will be needed in making the combined files, so we need to extract it from our stringtie generated gene\_abund.tab files.

First generate separate FPKM matrices

# Read in the gene\_abund.tab files, and subset a data frame of Gene ID, Gene Name and FPKM  
library(readr)  
  
A1\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\A1\_stringtie\\A1gene\_abund.tab", "\t")  
A1\_FPKMs <- subset(A1\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
A2\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\A2\_stringtie\\A2gene\_abund.tab", "\t")  
A2\_FPKMs <- subset(A2\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
A3\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\A3\_stringtie\\A3gene\_abund.tab", "\t")  
A3\_FPKMs <- subset(A3\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
A4\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\A4\_stringtie\\A4gene\_abund.tab", "\t")  
A4\_FPKMs <- subset(A4\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
B1\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\B1\_stringtie\\B1gene\_abund.tab", "\t")  
B1\_FPKMs <- subset(B1\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
B2\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\B2\_stringtie\\B2gene\_abund.tab", "\t")  
B2\_FPKMs <- subset(B2\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
B3\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\B3\_stringtie\\B3gene\_abund.tab", "\t")  
B3\_FPKMs <- subset(B3\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
B4\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\B4\_stringtie\\B4gene\_abund.tab", "\t")  
B4\_FPKMs <- subset(B4\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
C1\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\C1\_stringtie\\C1gene\_abund.tab", "\t")  
C1\_FPKMs <- subset(C1\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
C2\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\C2\_stringtie\\C2gene\_abund.tab", "\t")  
C2\_FPKMs <- subset(C2\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
C3\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\C3\_stringtie\\C3gene\_abund.tab", "\t")  
C3\_FPKMs <- subset(C3\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))  
  
C4\_FPKMs <- read\_delim("C:\\Users\\2060576e\\Documents\\C4\_stringtie\\C4gene\_abund.tab", "\t")  
C4\_FPKMs <- subset(C4\_FPKMs, select=c("Gene ID", "Start", "End", "FPKM"))

Then merge the matrices together to form one FPKM matrix

# Make the FPKM column names unique  
colnames(A1\_FPKMs)[4] <- "A1"  
colnames(A2\_FPKMs)[4] <- "A2"  
colnames(A3\_FPKMs)[4] <- "A3"  
colnames(A4\_FPKMs)[4] <- "A4"  
  
colnames(B1\_FPKMs)[4] <- "B1"  
colnames(B2\_FPKMs)[4] <- "B2"  
colnames(B3\_FPKMs)[4] <- "B3"  
colnames(B4\_FPKMs)[4] <- "B4"  
  
colnames(C1\_FPKMs)[4] <- "C1"  
colnames(C2\_FPKMs)[4] <- "C2"  
colnames(C3\_FPKMs)[4] <- "C3"  
colnames(C4\_FPKMs)[4] <- "C4"  
  
# Vector of column names to merge the files  
GeneInfo <- c("Gene ID", "Start", "End")  
  
  
# Merging the files  
FPKMs <- merge(A1\_FPKMs, A2\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, A3\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, A4\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, B1\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, B2\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, B3\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, B4\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, C1\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, C2\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, C3\_FPKMs, by=GeneInfo, sort=FALSE)  
FPKMs <- merge(FPKMs, C4\_FPKMs, by=GeneInfo, sort=FALSE)  
  
  
# Save FPKM matrix as a csv file  
library(readr)  
write\_csv(FPKMs, "C:\\Users\\2060576e\\Documents\\mouse\_chr2\_FPKM\_matrix.csv", append=FALSE)

Annotate and combine DE analysis files, and FPKM matrix, together. This will create a master file.

# Read in the annotation data  
annotations <- read.table("mart\_export.txt", header=TRUE, sep="\t")  
  
# Read in the FPKM matrix  
FPKMs <- read.csv("mouse\_chr2\_FPKM\_matrix.csv", header=TRUE)  
  
# Read in the DE analysis files  
A\_vs\_B <- read.csv("A\_vs\_B.csv", header=TRUE)  
B\_vs\_C <- read.csv("B\_vs\_C.csv", header=TRUE)  
A\_vs\_C <- read.csv("A\_vs\_C.csv", header=TRUE)  
  
  
# Make column names unique to each DE analysis data set  
colnames(A\_vs\_B) <- c("Gene.ID","AvsB\_BaseMean","AvsB\_log2FoldChange","AvsB\_lfcSE","AvsB\_stat","AvsB\_pvalue","AvsB\_padj")  
  
colnames(B\_vs\_C) <- c("Gene.ID","BvsC\_BaseMean","BvsC\_log2FoldChange","BvsC\_lfcSE","BvsC\_stat","BvsC\_pvalue","BvsC\_padj")  
  
colnames(A\_vs\_C) <- c("Gene.ID","AvsC\_BaseMean","AvsC\_log2FoldChange","AvsC\_lfcSE","AvsC\_stat","AvsC\_pvalue","AvsC\_padj")  
  
  
# Change column names in annotation data to match columns in FPKM data  
colnames(annotations) <- c("ID","Gene Name","Chromosome.scaffold.name","Start","End","Gene.type")  
  
# Vector of column names to merge the files  
colInfo <- c("Start", "End")  
  
# Merge the files to create master file  
FPKMs\_annot <- merge(FPKMs, annotations, by=colInfo)  
merged\_DEs <- merge(A\_vs\_B, B\_vs\_C, by="Gene.ID")  
merged\_DEs <- merge(merged\_DEs, A\_vs\_C, by="Gene.ID")  
masterFile <- merge(FPKMs\_annot, merged\_DEs, by="Gene.ID")

Reorder columns in master file, and remove redundant columns

# Remove the Gene.ID column from the master file  
## Becasue we already have another ID column provided by the annotation file  
#discard <- "Gene.ID"  
masterFile <- masterFile[ , -c(1)]  
  
# Reorder columns so that genomic identifiaction information are the first set of columns  
masterFile <- masterFile[c(15, 16, 17, 18, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36)]  
  
  
# Save data to file  
write\_delim(masterFile, "C:\\Users\\2060576e\\Documents\\Master\_File.txt", "\t", append=FALSE)

## 1. Checking for variation in measurement technique by plotting the distribution of expression values for each sample.

# Read in the master file:  
library(ggplot2)  
library(ggfortify)  
masterFile <- read\_delim("C:\\Users\\2060576e\\Documents\\Master\_File.txt", "\t")  
  
#We generate the plot:  
## Distribution plot for A1  
plot1 <- ggplot(masterFile, aes(x=log10(masterFile$A1))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("A1") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for A2  
plot2 <- ggplot(masterFile, aes(x=log10(masterFile$A2))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("A2") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for A3  
plot3 <- ggplot(masterFile, aes(x=log10(masterFile$A3))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("A3") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for A4  
plot4 <- ggplot(masterFile, aes(x=log10(masterFile$A4))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("A4") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for B1  
plot5 <- ggplot(masterFile, aes(x=log10(masterFile$B1))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("B1") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for B2  
plot6 <- ggplot(masterFile, aes(x=log10(masterFile$B2))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("B2") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for B3  
plot7 <- ggplot(masterFile, aes(x=log10(masterFile$B3))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("B3") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for B4  
plot8 <- ggplot(masterFile, aes(x=log10(masterFile$B4))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("B4") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for C1  
plot9 <- ggplot(masterFile, aes(x=log10(masterFile$C1))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("C1") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for C2  
plot10 <- ggplot(masterFile, aes(x=log10(masterFile$C2))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("C2") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for C3  
plot11 <- ggplot(masterFile, aes(x=log10(masterFile$C3))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("C3") +  
 theme\_bw() +  
 theme(legend.position="none")  
  
## Distribution plot for C4  
plot12 <- ggplot(masterFile, aes(x=log10(masterFile$C4))) +  
 geom\_density(fill="lightblue") +  
 xlim(-5,5) +  
 xlab("Expression (log10)") +  
 ylab("Density") +  
 ggtitle("C4") +  
 theme\_bw() +  
 theme(legend.position="none")

Make a multiplot of the distibution of expression values

# Generate multiplot using multiplot function, and save as a png  
png("Mouse\_DistExprs.png", height=750, width=750, pointsize=10)  
p <- multiplot(plot1,plot5,plot9,plot2,plot6,plot10,plot3,plot7,plot11,plot4,plot8,plot12, cols= 4)  
print(p)  
dev.off()

## 2.Visualising the distribution of log2fold changes across mean expression levels

This plot is useful for investigating whether the upregulation (or downregulation) of genes is more or less than expected. The name of such a plot is called an MA plot.

# Read in the master file. Make the ID columns the rownames  
masterFile <- read.table("Master\_File.txt", header=TRUE, sep = "\t", row.names = 1)  
  
  
# First create a separate table of only expression values  
FPKM\_columns <- c("A1", "A2", "A3", "A4", "B1", "B2", "B3", "B4", "C1", "C2", "C3", "C4")  
mfFPKMs <- masterFile[,FPKM\_columns]  
  
# Generate mean expression column and add it to master file  
masterFile$row\_means <- rowMeans(mfFPKMs, na.rm = FALSE, dims = 1)  
  
# Generate and add a column that flags significance as "true"" or "false" to the masterFile  
### This will require the p-adjusted column of each DE analysis  
### The threshold for the p-adjusted value is 0.05   
masterFile$AvsB\_sigFlag <- as.factor(masterFile$AvsB\_padj > 0.05|abs(masterFile$AvsB\_log2FoldChange)<1.0)  
masterFile$BvsC\_sigFlag <- as.factor(masterFile$BvsC\_padj > 0.05|abs(masterFile$BvsC\_log2FoldChange)<1.0)  
masterFile$AvsC\_sigFlag <- as.factor(masterFile$AvsC\_padj > 0.05|abs(masterFile$AvsC\_log2FoldChange)<1.0)  
  
  
# Make list of significant genes in each DE analysis  
AvsB\_sigGenes <- subset(masterFile, masterFile$AvsB\_padj<0.05 & abs(masterFile$AvsB\_log2FoldChange)>1.0)  
BvsC\_sigGenes <- subset(masterFile, masterFile$BvsC\_padj<0.05 & abs(masterFile$BvsC\_log2FoldChange)>1.0)  
AvsC\_sigGenes <- subset(masterFile, masterFile$AvsC\_padj<0.05 & abs(masterFile$AvsC\_log2FoldChange)>1.0)  
  
# Save the significant genes master file to an output file  
write\_delim(AvsB\_sigGenes, "C:\\Users\\2060576e\\Documents\\AvsB\_sigGenes.txt", "\t", append=FALSE)  
write\_delim(BvsC\_sigGenes, "C:\\Users\\2060576e\\Documents\\BvsC\_sigGenes.txt", "\t", append=FALSE)  
write\_delim(AvsC\_sigGenes, "C:\\Users\\2060576e\\Documents\\AvsC\_sigGenes.txt", "\t", append=FALSE)

Generate M(log ratio) - A(mean expression level) plot for each DE analysis

# MA plots with dots by coloured by significance.  
### Only significant genes will be labelled  
png("AB\_MA.png", width = 600, height = 550, pointsize=5)  
ggplot(data=masterFile, aes(x=log10(masterFile$row\_means+0.001), y=masterFile$AvsB\_log2FoldChange, colour =masterFile$AvsB\_sigFlag)) +   
 geom\_point(size=1.8) +   
 geom\_hline(aes(yintercept = 0), colour = "black", size = 0.75) +   
 xlab("Log2 Mean Expression") +   
 ylab("Log2 Fold Change") +   
 theme\_bw() +   
 theme(legend.position="bottom",legend.title = element\_blank()) +   
 scale\_colour\_discrete(breaks=c("FALSE","TRUE"),labels=c("Significant", "Not-significant"))  
dev.off()  
  
png("BC\_MA.png", width = 600, height = 550, pointsize=5)  
ggplot(data=masterFile, aes(x=log10(masterFile$row\_means+0.001), y=masterFile$BvsC\_log2FoldChange, colour =masterFile$BvsC\_sigFlag)) +   
 geom\_point(size=1.8) +   
 geom\_hline(aes(yintercept = 0), colour = "black", size = 0.75) +   
 xlab("Log2 Mean Expression") +   
 ylab("Log2 Fold Change") +   
 theme\_bw() +   
 theme(legend.position="bottom",legend.title = element\_blank()) +   
 scale\_colour\_discrete(breaks=c("FALSE","TRUE"),labels=c("Significant", "Not-significant"))  
dev.off()  
  
png("AC\_MA.png", width = 600, height = 550, pointsize=5)  
ggplot(data=masterFile, aes(x=log10(masterFile$row\_means+0.001), y=masterFile$AvsC\_log2FoldChange, colour =masterFile$AvsC\_sigFlag)) +   
 geom\_point(size=1.8) +   
 geom\_hline(aes(yintercept = 0), colour = "black", size = 0.75) +   
 xlab("Log2 Mean Expression") +   
 ylab("Log2 Fold Change") +   
 theme\_bw() +   
 theme(legend.position="bottom",legend.title = element\_blank()) +   
 scale\_colour\_discrete(breaks=c("FALSE","TRUE"),labels=c("Significant", "Not-significant"))  
dev.off()

## 3. Observing whether significantly differentially expressed genes show consistent expression patterns within replicates with a clustered Image map

Giving that we are working with only significant genes, we have to first generate a master file that contains only significant genes

# Read in master files with significantly expressed genes  
AvsB\_sigGenes <- read.table("AvsB\_sigGenes.txt", header=TRUE, sep = "\t", row.names = 1)  
BvsC\_sigGenes <- read.table("BvsC\_sigGenes.txt", header=TRUE, sep= "\t", row.names=1)  
AvsC\_sigGenes <- read.table("AvsC\_sigGenes.txt", header=TRUE, sep= "\t", row.names=1)  
  
# To prevent crashing due to high memory usage, a code is implemented to restrict the number of rows to 5000  
  
  
if(nrow(AvsB\_sigGenes) > 5000){  
 AvsB\_sigGenes <- AvsB\_sigGenes[sample(1:nrow(AvsB\_sigGenes), 5000, replace = FALSE),]  
}  
if(nrow(BvsC\_sigGenes) > 5000){  
 BvsC\_sigGenes <- BvsC\_sigGenes[sample(1:nrow(BvsC\_sigGenes), 5000, replace = FALSE),]  
}  
if(nrow(AvsC\_sigGenes) > 5000){  
 AvsC\_sigGenes <- AvsC\_sigGenes[sample(1:nrow(AvsC\_sigGenes), 5000, replace = FALSE),]  
}  
  
  
  
  
# Select only the columns with the expression values  
FPKM\_columnsAB <- c("A1", "A2", "A3", "A4", "B1", "B2", "B3", "B4")  
FPKM\_columnsBC <- c("B1", "B2", "B3", "B4", "C1", "C2", "C3", "C4")  
FPKM\_columnsAC <- c("A1", "A2", "A3", "A4", "C1", "C2", "C3", "C4")  
  
AvsB\_sigGenes <- AvsB\_sigGenes[, FPKM\_columnsAB]  
BvsC\_sigGenes <- BvsC\_sigGenes[, FPKM\_columnsBC]  
AvsC\_sigGenes <- AvsC\_sigGenes[, FPKM\_columnsAC]

Working with transformed versions of count data is advisable for clustering. It allows us to see significant differential expression in genes with low expression values

# Scale the expression values for each gene using the Z-scores  
AvsB\_sigGenes.scale <- t(scale(t(AvsB\_sigGenes)))  
BvsC\_sigGenes.scale <- t(scale(t(BvsC\_sigGenes)))  
AvsC\_sigGenes.scale <- t(scale(t(AvsC\_sigGenes)))  
  
  
# Cluster rows of similar expression levels  
library(amap)  
row.order <- hclust(Dist(AvsB\_sigGenes.scale, method ="spearman"), method = "average")$order  
row.order1 <- hclust(Dist(BvsC\_sigGenes.scale, method ="spearman"), method = "average")$order  
#row.order2 <- hclust(Dist(AvsC\_sigGenes.scale, method ="spearman"), method = "average")$order  
  
AvsB\_sigGenes.scale.clustered <- AvsB\_sigGenes.scale[row.order,]  
BvsC\_sigGenes.scale.clustered <- BvsC\_sigGenes.scale[row.order1,]  
#AvsC\_sigGenes.scale.clustered <- AvsC\_sigGenes.scale[row.order2,]  
  
  
  
# Convert data frame to format that is compatible with ggplot  
library(reshape2)  
AvsB\_sigGenes.m <- melt(as.matrix(AvsB\_sigGenes.scale.clustered))  
BvsC\_sigGenes.m <- melt(as.matrix(BvsC\_sigGenes.scale.clustered))  
AvsC\_sigGenes.m <- melt(as.matrix(AvsC\_sigGenes.scale))  
  
# Generate the heatmaps  
hm.palette <- colorRampPalette(c("magenta", "black", "yellow"))  
  
png("A-B\_heat.png", width = 500, height = 500, pointsize=10)  
ggplot(AvsB\_sigGenes.m, aes(x = Var2, y = Var1, fill = value)) + geom\_tile() +   
 ggtitle("A vs B - Significant Genes") +   
 scale\_fill\_gradientn(colours=hm.palette(100), name="Row Z-score") +   
 ylab('Genes') + xlab('Samples') +  
 theme\_bw() +   
 theme(axis.text.x=element\_text(angle=90, hjust = 1), axis.ticks.x = element\_blank(), axis.ticks.length = unit(0.5, "cm"), plot.background = element\_blank(),panel.grid.major = element\_blank(), panel.grid.minor = element\_blank(),panel.border = element\_blank())  
dev.off()  
  
png("B-C\_heat.png", width = 500, height = 500, pointsize=10)  
ggplot(BvsC\_sigGenes.m, aes(x = Var2, y = Var1, fill = value)) + geom\_tile() +   
 ggtitle("B vs C - Significant Genes") +   
 scale\_fill\_gradientn(colours=hm.palette(100), name="Row Z-score") +   
 ylab('Genes') + xlab('Samples') +  
 theme\_bw() +   
 theme(axis.text.x=element\_text(angle=90, hjust = 1), axis.ticks.x = element\_blank(), axis.ticks.length = unit(0.5, "cm"), plot.background = element\_blank(),panel.grid.major = element\_blank(), panel.grid.minor = element\_blank(),panel.border = element\_blank())  
dev.off()  
  
png("A-C\_heat.png", width = 500, height = 100, pointsize=5)  
ggplot(AvsC\_sigGenes.m, aes(x = Var2, y = Var1, fill = value)) + geom\_tile() +   
 ggtitle("A vs C - Significant Genes") +   
 scale\_fill\_gradientn(colours=hm.palette(100), name="Row Z-score") +   
 ylab('Genes') + xlab('Samples') +  
 theme\_bw() +   
 theme(axis.text.x=element\_text(angle=90, hjust = 1), axis.ticks.x = element\_blank(), axis.ticks.length = unit(0.5, "cm"), plot.background = element\_blank(),panel.grid.major = element\_blank(), panel.grid.minor = element\_blank(),panel.border = element\_blank())  
dev.off()

## 4. Observing whether non-significantly differentially expressed genes show consistent expression patterns within replicates with a clustered Image map

Do our non-significant genes look really non-significant?

# Read in the master file  
masterFile <- read.table("Master\_File.txt", header=TRUE, sep = "\t", row.names = 2)  
  
# Make master files of non-significant genes in each DE analysis  
AvsB\_Non\_sigGenes <- subset(masterFile, masterFile$AvsB\_padj>0.05 & abs(masterFile$AvsB\_log2FoldChange)<1.0)  
BvsC\_Non\_sigGenes <- subset(masterFile, masterFile$BvsC\_padj>0.05 & abs(masterFile$BvsC\_log2FoldChange)<1.0)  
AvsC\_Non\_sigGenes <- subset(masterFile, masterFile$AvsC\_padj>0.05 & abs(masterFile$AvsC\_log2FoldChange)<1.0)  
  
  
# To prevent crashing due to high memory usage, a code is implemented to restrict the number of rows to 5000  
  
  
if(nrow(AvsB\_Non\_sigGenes) > 5000){  
 AvsB\_Non\_sigGenes <- AvsB\_Non\_sigGenes[sample(1:nrow(AvsB\_Non\_sigGenes), 5000, replace = FALSE),]  
}  
if(nrow(BvsC\_Non\_sigGenes) > 5000){  
 BvsC\_Non\_sigGenes <- BvsC\_Non\_sigGenes[sample(1:nrow(BvsC\_Non\_sigGenes), 5000, replace = FALSE),]  
}  
if(nrow(AvsC\_Non\_sigGenes) > 5000){  
 AvsC\_Non\_sigGenes <- AvsC\_Non\_sigGenes[sample(1:nrow(AvsC\_Non\_sigGenes), 5000, replace = FALSE),]  
}  
  
  
  
  
# Select only the columns with the expression values  
FPKM\_columnsAB <- c("A1", "A2", "A3", "A4", "B1", "B2", "B3", "B4")  
FPKM\_columnsBC <- c("B1", "B2", "B3", "B4", "C1", "C2", "C3", "C4")  
FPKM\_columnsAC <- c("A1", "A2", "A3", "A4", "C1", "C2", "C3", "C4")  
  
AvsB\_Non\_sigGenes <- AvsB\_Non\_sigGenes[, FPKM\_columnsAB]  
BvsC\_Non\_sigGenes <- BvsC\_Non\_sigGenes[, FPKM\_columnsBC]  
AvsC\_Non\_sigGenes <- AvsC\_Non\_sigGenes[, FPKM\_columnsAC]

Working with transformed versions of count data is advisable for clustering. It allows us to see significant differential expression in genes with low expression values

# Scale the expression values for each gene using the Z-scores  
AvsB\_Non\_sigGenes.scale <- t(scale(t(AvsB\_Non\_sigGenes)))  
BvsC\_Non\_sigGenes.scale <- t(scale(t(BvsC\_Non\_sigGenes)))  
AvsC\_Non\_sigGenes.scale <- t(scale(t(AvsC\_Non\_sigGenes)))  
  
  
# Cluster rows of similar expression levels  
library(amap)  
row.order <- hclust(Dist(AvsB\_Non\_sigGenes.scale, method ="spearman"), method = "average")$order  
row.order1 <- hclust(Dist(BvsC\_Non\_sigGenes.scale, method ="spearman"), method = "average")$order  
row.order2 <- hclust(Dist(AvsC\_Non\_sigGenes.scale, method ="spearman"), method = "average")$order  
  
AvsB\_Non\_sigGenes.scale.clustered <- AvsB\_Non\_sigGenes.scale[row.order,]  
BvsC\_Non\_sigGenes.scale.clustered <- BvsC\_Non\_sigGenes.scale[row.order1,]  
AvsC\_Non\_sigGenes.scale.clustered <- AvsC\_Non\_sigGenes.scale[row.order2,]  
  
  
  
# Convert data frame to format that is compatible with ggplot  
library(reshape2)  
AvsB\_Non\_sigGenes.m <- melt(as.matrix(AvsB\_Non\_sigGenes.scale.clustered))  
BvsC\_Non\_sigGenes.m <- melt(as.matrix(BvsC\_Non\_sigGenes.scale.clustered))  
AvsC\_Non\_sigGenes.m <- melt(as.matrix(AvsC\_Non\_sigGenes.scale))  
  
# Generate the heatmaps  
hm.palette <- colorRampPalette(c("magenta", "black", "yellow"))  
  
  
png("NonSig\_AB-heat.png", width = 500, height = 600, pointsize=5)  
ggplot(AvsB\_Non\_sigGenes.m, aes(x = Var2, y = Var1, fill = value)) + geom\_tile() +   
 ggtitle("A vs B - Non Significant Genes") +   
 scale\_fill\_gradientn(colours=hm.palette(100), name="Row Z-score") +   
 ylab('Genes') + xlab('Samples') +  
 theme\_bw() +   
 theme(axis.text.x=element\_text(angle=90, hjust = 1), axis.ticks.x = element\_blank(), axis.ticks.length = unit(0.5, "cm"), plot.background = element\_blank(),panel.grid.major = element\_blank(), panel.grid.minor = element\_blank(),panel.border = element\_blank())  
dev.off()  
  
png("NonSig\_BC-heat.png", width = 500, height = 600, pointsize=5)  
ggplot(BvsC\_Non\_sigGenes.m, aes(x = Var2, y = Var1, fill = value)) + geom\_tile() +   
 ggtitle("B vs C - Non Significant Genes") +   
 scale\_fill\_gradientn(colours=hm.palette(100), name="Row Z-score") +   
 ylab('Genes') + xlab('Samples') +  
 theme\_bw() +   
 theme(axis.text.x=element\_text(angle=90, hjust = 1), axis.ticks.x = element\_blank(), axis.ticks.length = unit(0.5, "cm"), plot.background = element\_blank(),panel.grid.major = element\_blank(), panel.grid.minor = element\_blank(),panel.border = element\_blank())  
dev.off()  
  
png("NonSig\_AC-heat.png", width = 500, height = 600, pointsize=5)  
ggplot(AvsC\_Non\_sigGenes.m, aes(x = Var2, y = Var1, fill = value)) + geom\_tile() +   
 ggtitle("A vs C - Non Significant Genes") +   
 scale\_fill\_gradientn(colours=hm.palette(100), name="Row Z-score") +   
 ylab('Genes') + xlab('Samples') +  
 theme\_bw() +   
 theme(axis.text.x=element\_text(angle=90, hjust = 1), axis.ticks.x = element\_blank(), axis.ticks.length = unit(0.5, "cm"), plot.background = element\_blank(),panel.grid.major = element\_blank(), panel.grid.minor = element\_blank(),panel.border = element\_blank())  
dev.off()

## 5. Bar charts of the number of significantly up and down-regulated genes

# Get the number of significantly upregulated genes  
masterFile <- read\_delim("C:\\Users\\2060576e\\Documents\\Master\_File.txt", "\t")  
AB\_upregulated\_count <- nrow(subset(masterFile, masterFile$AvsB\_padj<0.05 & masterFile$AvsB\_log2FoldChange>1.0))  
  
BC\_upregulated\_count <- nrow(subset(masterFile, masterFile$BvsC\_padj<0.05 & masterFile$BvsC\_log2FoldChange>1.0))  
  
AC\_upregulated\_count <- nrow(subset(masterFile, masterFile$AvsC\_padj<0.05 & masterFile$AvsC\_log2FoldChange>1.0))  
  
tot\_upregulated\_count <- AB\_upregulated\_count + BC\_upregulated\_count + AC\_upregulated\_count  
  
  
# Get the number significantly downregulated genes  
AB\_downregulated\_count <- nrow(subset(masterFile, masterFile$AvsB\_padj<0.05 & masterFile$AvsB\_log2FoldChange< -1.0))  
  
BC\_downregulated\_count <- nrow(subset(masterFile, masterFile$BvsC\_padj<0.05 & masterFile$BvsC\_log2FoldChange< -1.0))  
  
AC\_downregulated\_count <- nrow(subset(masterFile, masterFile$AvsC\_padj<0.05 & masterFile$AvsC\_log2FoldChange< -1.0))  
  
tot\_downregulated\_count <- AB\_downregulated\_count + BC\_downregulated\_count + AC\_downregulated\_count  
  
  
# Make table of results for bar plot  
### Make column that descibes the comparison  
comparison <- c("A\_vs\_B", "A\_vs\_B", "B\_vs\_C", "B\_vs\_C", "A\_vs\_C", "A\_vs\_C")  
  
### Make column that describes the direction of expression  
direction <- c("upregulated", "downregualted", "upregulated", "downregualted", "upregulated", "downregualted")  
  
### Make column of the number of upregualted or downregulated genes  
number\_of\_sig\_genes <- c(AB\_upregulated\_count, AB\_downregulated\_count, BC\_upregulated\_count, BC\_downregulated\_count, AC\_upregulated\_count, AC\_downregulated\_count )  
  
dat <- data.frame(comparison, direction, number\_of\_sig\_genes)  
  
  
# Plot the bar chart  
png("Bar\_plot.png", width = 600, height = 500, pointsize=10)  
ggplot(data=dat , aes(x=comparison, y=number\_of\_sig\_genes, fill=direction)) + geom\_bar(colour="black", stat="identity", position =  
"dodge") + ylab("Number of significant genes") + xlab("Comparison") + scale\_fill\_discrete(name = "Direction") + theme\_bw() + geom\_text(aes(label=number\_of\_sig\_genes), position=position\_dodge(width=0.9), vjust=2)  
dev.off()

## 6. Violin and jitter plots of the top differential genes

First extract the significant DE genes from the sigGenes master file, then order them by fold change

# Read in master file of significant DE genes  
AvsB\_sigGenes <- read\_delim("C:\\Users\\2060576e\\Documents\\AvsB\_sigGenes.txt", "\t")  
BvsC\_sigGenes <- read\_delim("C:\\Users\\2060576e\\Documents\\BvsC\_sigGenes.txt", "\t")  
AvsC\_sigGenes <- read\_delim("C:\\Users\\2060576e\\Documents\\AvsC\_sigGenes.txt", "\t")  
  
# Sort rows by p values in ascending order  
AB\_Sig.ordered <- AvsB\_sigGenes[order(AvsB\_sigGenes$AvsB\_padj, decreasing = FALSE),]  
BC\_Sig.ordered <- BvsC\_sigGenes[order(BvsC\_sigGenes$BvsC\_padj, decreasing = FALSE),]  
AC\_Sig.ordered <- AvsC\_sigGenes[order(AvsC\_sigGenes$AvsC\_padj, decreasing = FALSE),]  
  
# Transform to data frames  
AB\_Sig.ordered <- as.data.frame(AB\_Sig.ordered)  
BC\_Sig.ordered <- as.data.frame(BC\_Sig.ordered)  
AC\_Sig.ordered <- as.data.frame(AC\_Sig.ordered)  
  
# Top three upregulated genes   
### For A vs B  
AB\_gene1 <- AB\_Sig.ordered[1,1]  
AB\_gene2 <- AB\_Sig.ordered[2,1]  
AB\_gene3 <- AB\_Sig.ordered[3,1]  
  
### For B vs C  
BC\_gene1 <- BC\_Sig.ordered[1,1]  
BC\_gene2 <- BC\_Sig.ordered[2,1]  
BC\_gene3 <- BC\_Sig.ordered[3,1]  
  
### A vs C has no upregulated genes  
  
# Top three downregulated genes  
### For A vs B  
AB\_Un\_gene1 <- AB\_Sig.ordered[8,1]  
AB\_Un\_gene2 <- AB\_Sig.ordered[7,1]  
AB\_Un\_gene3 <- AB\_Sig.ordered[6,1]  
  
### For B vs C  
BC\_Un\_gene1 <- BC\_Sig.ordered[10,1]  
BC\_Un\_gene2 <- BC\_Sig.ordered[9,1]  
BC\_Un\_gene3 <- BC\_Sig.ordered[8,1]  
  
### For A vs C  
AC\_Un\_gene1 <- AC\_Sig.ordered[1,1]

Next, we need to format our ordered master table into a table that ggplot will accept

# Comma separated list of FPKM columns  
FP1 <- c("A1", "A2", "A3", "A4", "B1", "B2", "B3", "B4")  
FP2 <- c("B1", "B2", "B3", "B4", "C1", "C2", "C3", "C4")  
FP3 <- c("A1", "A2", "A3", "A4", "C1", "C2", "C3", "C4")  
  
  
# Trim the tables, so that table only contains top 3 rows (For top 3 upregulated genes)  
# Also trim unnecesaary column  
#Also transform the tables  
AB\_Sig.ordered.up <- AB\_Sig.ordered[c(1,2,3),]  
AB\_Sig.ordered.up <- AB\_Sig.ordered.up[,FP1]  
AB\_Sig.ordered.up <- t(AB\_Sig.ordered.up)  
AB\_Sig.ordered.up <- as.data.frame(AB\_Sig.ordered.up)  
  
BC\_Sig.ordered.up <- BC\_Sig.ordered.up <- BC\_Sig.ordered[c(1,2,3),]  
BC\_Sig.ordered.up <- BC\_Sig.ordered.up <- BC\_Sig.ordered.up[,FP2]  
BC\_Sig.ordered.up <- t(BC\_Sig.ordered.up)  
BC\_Sig.ordered.up <- as.data.frame(BC\_Sig.ordered.up)  
  
  
# Trim the tables, so that table only contains bottom 3 rows (For top 3 downregulated genes)  
# Also trim unnecesaary column  
#Also transform the tables  
AB\_Sig.ordered.down <- AB\_Sig.ordered[c(8,7,6),]  
AB\_Sig.ordered.down <- AB\_Sig.ordered.down[,FP1]  
AB\_Sig.ordered.down <- t(AB\_Sig.ordered.down)  
AB\_Sig.ordered.down <- as.data.frame(AB\_Sig.ordered.down)  
  
BC\_Sig.ordered.down <- BC\_Sig.ordered[c(10,9,8),]  
BC\_Sig.ordered.down <- BC\_Sig.ordered.down[,FP2]  
BC\_Sig.ordered.down <- t(BC\_Sig.ordered.down)  
BC\_Sig.ordered.down <- as.data.frame(BC\_Sig.ordered.down)  
  
AC\_Sig.ordered.down <- AC\_Sig.ordered[,FP3]  
AC\_Sig.ordered.down <- t(AC\_Sig.ordered.down)  
AC\_Sig.ordered.down <- as.data.frame(AC\_Sig.ordered.down)  
  
  
# Append sample group columns  
AB\_Sig.ordered.up$sample\_groups <- c(rep("A", 4), rep("A\_vs\_B", 4))  
BC\_Sig.ordered.up$sample\_groups <- c(rep("B", 4), rep("B\_vs\_C", 4))  
AB\_Sig.ordered.down$sample\_groups <- c(rep("A", 4), rep("A\_vs\_B", 4))  
BC\_Sig.ordered.down$sample\_groups <- c(rep("B", 4), rep("B\_vs\_C", 4))  
AC\_Sig.ordered.down$sample\_groups <- c(rep("A", 4), rep("A\_vs\_C", 4))

Generate the plot for the top 3 differentially expressed genes in A vs B

# Plot the top 3 upregulated genes for A vs B  
  
plot1 <- ggplot(AB\_Sig.ordered.up, aes(x=AB\_Sig.ordered.up$sample\_groups,y=AB\_Sig.ordered.up[,1], colour=AB\_Sig.ordered.up$sample\_groups, fill=AB\_Sig.ordered.up$sample\_groups)) +  
geom\_jitter(size=1, colour="black", width=0.25) + geom\_violin(trim=FALSE) + xlab("") + ylab("Expression level") + ggtitle(AB\_gene1) + stat\_summary(position=position\_dodge(0.75), colour = "red", size = 0.25, geom="pointrange") + theme\_bw() + theme(legend.position="none",plot.title = element\_text(size=12))  
  
plot2 <- ggplot(AB\_Sig.ordered.up, aes(x=AB\_Sig.ordered.up$sample\_groups,y=AB\_Sig.ordered.up[,2], colour=AB\_Sig.ordered.up$sample\_groups, fill=AB\_Sig.ordered.up$sample\_groups)) +  
geom\_jitter(size=1, colour="black", width=0.25) + geom\_violin(trim=FALSE) + xlab("") + ylab("Expression level") + ggtitle(AB\_gene2) + stat\_summary(position=position\_dodge(0.75), colour = "red", size = 0.25, geom="pointrange") + theme\_bw() + theme(legend.position="none",plot.title = element\_text(size=12))  
  
plot3 <- ggplot(AB\_Sig.ordered.up, aes(x=AB\_Sig.ordered.up$sample\_groups,y=AB\_Sig.ordered.up[,3], colour=AB\_Sig.ordered.up$sample\_groups, fill=AB\_Sig.ordered.up$sample\_groups)) +  
geom\_jitter(size=1, colour="black", width=0.25) + geom\_violin(trim=FALSE) + xlab("") + ylab("Expression level") + ggtitle(AB\_gene3) + stat\_summary(position=position\_dodge(0.75), colour = "red", size = 0.25, geom="pointrange") + theme\_bw() + theme(legend.position="none",plot.title = element\_text(size=12))  
  
png("AB-upreg.png", height=250, width=900, pointsize=10)  
p <- multiplot(plot1,plot2,plot3, cols= 3)  
print(p)  
dev.off()

Generate the plot for the top 3 differentially expressed genes in B vs C

# Plot the top 3 upregulated genes for B vs C  
  
plot1 <- ggplot(BC\_Sig.ordered.up, aes(x=BC\_Sig.ordered.up$sample\_groups,y=BC\_Sig.ordered.up[,1], colour=BC\_Sig.ordered.up$sample\_groups, fill=BC\_Sig.ordered.up$sample\_groups)) +  
geom\_jitter(size=1, colour="black", width=0.25) + geom\_violin(trim=FALSE) + xlab("") + ylab("Expression level") + ggtitle(BC\_gene1) + stat\_summary(position=position\_dodge(0.75), colour = "red", size = 0.25, geom="pointrange") + theme\_bw() + theme(legend.position="none",plot.title = element\_text(size=12))  
  
plot2 <- ggplot(BC\_Sig.ordered.up, aes(x=BC\_Sig.ordered.up$sample\_groups,y=BC\_Sig.ordered.up[,2], colour=BC\_Sig.ordered.up$sample\_groups, fill=BC\_Sig.ordered.up$sample\_groups)) +  
geom\_jitter(size=1, colour="black", width=0.25) + geom\_violin(trim=FALSE) + xlab("") + ylab("Expression level") + ggtitle(BC\_gene2) + stat\_summary(position=position\_dodge(0.75), colour = "red", size = 0.25, geom="pointrange") + theme\_bw() + theme(legend.position="none",plot.title = element\_text(size=12))  
  
plot3 <- ggplot(BC\_Sig.ordered.up, aes(x=BC\_Sig.ordered.up$sample\_groups,y=BC\_Sig.ordered.up[,3], colour=BC\_Sig.ordered.up$sample\_groups, fill=BC\_Sig.ordered.up$sample\_groups)) +  
geom\_jitter(size=1, colour="black", width=0.25) + geom\_violin(trim=FALSE) + xlab("") + ylab("Expression level") + ggtitle(BC\_gene3) + stat\_summary(position=position\_dodge(0.75), colour = "red", size = 0.25, geom="pointrange") + theme\_bw() + theme(legend.position="none",plot.title = element\_text(size=12))  
  
png("BC-upreg.png", height=250, width=900, pointsize=10)  
p <- multiplot(plot1,plot2,plot3, cols= 3)  
print(p)  
dev.off()

Generate the plot for the differentially expressed gene in A vs C

png("AC-upreg.png", height=400, width=600)  
  
ggplot(AC\_Sig.ordered.down, aes(x=AC\_Sig.ordered.down$sample\_groups,y=AC\_Sig.ordered.down[,1], colour=AC\_Sig.ordered.down$sample\_groups, fill=AC\_Sig.ordered.down$sample\_groups)) +  
geom\_jitter(size=1, colour="black", width=0.25) + geom\_violin(trim=FALSE) + xlab("") + ylab("Expression level") + ggtitle(AC\_gene1) + stat\_summary(position=position\_dodge(0.75), colour = "red", size = 0.25, geom="pointrange") + theme\_bw() + theme(legend.position="none",plot.title = element\_text(size=12))  
  
  
dev.off()

## 7. PCA plot

# Read in the master file  
masterFile <- read.table("Master\_File.txt", header=TRUE, sep = "\t", row.names = 1)  
  
# Make list of FPKM columns  
FPKM\_columns <- c("A1", "A2", "A3", "A4", "B1", "B2", "B3", "B4", "C1", "C2", "C3", "C4")  
  
x <- masterFile[, FPKM\_columns]  
x <- log10(x+1)  
  
#We parse the expression matrix and perform PCA:  
x <- as.matrix(sapply(x, as.numeric))  
xx <- prcomp(t(x))  
  
# Make list of group names  
group <- c(rep("A", 4), rep("B", 4), rep("C", 4 ))  
  
#We plot the PCA scatter plot of PC1 vs PC2:  
png("R\_PCA.png", width = 500, height = 500, pointsize=5)  
y <- group  
ggplot(xx, aes(x=PC1,y=PC2, fill = group, colour = "Group")) + geom\_point(size=3, shape=21) +  
 geom\_text(aes(label=colnames(x)),hjust= 0.5, vjust=-1) +  
 ggtitle("Principal Component Analysis") +  
 theme\_bw() +  
 theme(legend.position="bottom", legend.title = element\_blank()) +  
 xlab(paste("PC1 (",summary(xx)$importance[2]\*100,"%)",sep="")) +  
 ylab(paste("PC2 (",summary(xx)$importance[5]\*100,"%)",sep=""))  
# ggplot(xx, aes(x=PC1,y=PC2)) + geom\_point(size=3, shape=21)  
  
dev.off()

## 8. Venn Diagram statistic for “A vs B” and “B vs C”

# Read in the annotation data and master file data  
annotations <- read.table("mart\_export.txt", header=TRUE, sep="\t")  
masterFile <- read\_delim("C:\\Users\\2060576e\\Documents\\Master\_File.txt", "\t")  
  
# Total number of genes in the genome  
gene\_total <- length(annotations$Associated.Gene.Name)  
  
# Number of upregulated genes in A vs B  
AB\_upregulated\_count <- nrow(subset(masterFile, masterFile$AvsB\_padj<0.05 & masterFile$AvsB\_log2FoldChange>1.0))  
  
# Number of upregulated genes in B vs C  
BC\_upregulated\_count <- nrow(subset(masterFile, masterFile$BvsC\_padj<0.05 & masterFile$BvsC\_log2FoldChange>1.0))  
  
# Number of downregulated genes in A vs B  
AB\_downregulated\_count <- nrow(subset(masterFile, masterFile$AvsB\_padj<0.05 & masterFile$AvsB\_log2FoldChange< -1.0))  
  
# Number of downregulated genes in B vs C  
BC\_downregulated\_count <- nrow(subset(masterFile, masterFile$BvsC\_padj<0.05 & masterFile$BvsC\_log2FoldChange< -1.0))  
  
# Number or overlapping genes in AB upreg vs BC downreg = 5  
# Number or overlapping genes in AB downreg vs BC upreg =   
  
# Run overlap statistics  
## For AB upreg vs BC downreg  
phyper(5, AB\_upregulated\_count, gene\_total - AB\_upregulated\_count, BC\_downregulated\_count, lower.tail = FALSE)  
  
  
## For AB downreg vs BC upreg  
phyper(1, AB\_downregulated\_count, gene\_total - AB\_downregulated\_count, BC\_upregulated\_count, lower.tail = FALSE)