library(limma)

library(DESeq2)

library(vsn)

library(ggplot2)

library(dplyr)

library(RColorBrewer)

library(pheatmap)

library(PoiClaClu)

library(edgeR)

library(DSS)

#Generate a table

gene\_data <- read.table("Genes.txt", h=T)

leftCounts = read.table(“LeftCounts.txt”, header = TRUE)

rightCounts = read.table(“RightCounts.txt”,header = TRUE)

#Check the data fields

dim(leftCounts)

dim(rightCounts)

dim(gene\_data)

counts\_data = cbind(leftCounts, rightCounts)

# Removes certain patients from the LEFT \_breast portion of the counts\_data object

counts\_data = subset(counts\_data, select = -c(X2e528e8d.46a5.42b0.aa58.d266b30190b0.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X1be91780.e668.4261.bd65.ee03f2a5a1e7.htseq.counts))

counts\_data = subset(counts\_data, select = -c(c0055307.e827.4621.82ff.9702ae53df10.htseq.counts))

counts\_data = subset(counts\_data, select = -c(d229a59b.4f6c.4714.8dd5.1c714a764b22.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X76f75946.4d85.415e.8602.d1153fe3e2c3.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X15f8a6a7.ab64.4106.b52f.601e3c5d92c5.htseq.counts))

counts\_data = subset(counts\_data, select = -c(e3a6fb76.0374.4823.91ec.7caaad95f1b9.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X856ca504.791f.45f1.b756.e180e6a2535a.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X36866b9a.180c.4d05.8030.6d0ace6f6adf.htseq.counts))

# Removes certain patients from the RIGHT \_breast portion of the counts\_data object

counts\_data = subset(counts\_data, select = -c(c56cda8a.0755.4715.9c46.5bc6280a5bf0.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X445b096b.502f.4e88.9338.4ad309a14425.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X46927791.e908.49eb.8ad8.21158fc1549a.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X33fef46f.c248.4d58.bb6e.3a4a55272334.htseq.counts))

counts\_data = subset(counts\_data, select = -c(b033c85a.2395.4ced.a84d.ac1fbce674bf.htseq.counts))

counts\_data = subset(counts\_data, select = -c(e7dd2f36.adee.4be2.ae3c.39d7664055e2.htseq.counts))

counts\_data = subset(counts\_data, select = -c(a4c7b6e8.ed30.42b2.9daa.0cf0b9d59259.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X2656413d.a5d2.4812.9ce9.a02c15ab04bd.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X8e21887e.f3ca.4c2c.819a.538952264c23.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X1d102e83.ede2.460e.ab2e.f42d8db6970e.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X20b241da.8e69.4eb0.a7f2.4eb24cf76059.htseq.counts))

counts\_data = subset(counts\_data, select = -c(X2fbb0bb8.38a2.48ef.a2d0.8fa2d01772b9.htseq.counts))

counts\_data = subset(counts\_data, select = -c(f4d80dc6.a28c.4d31.99e8.f58e78678654.htseq.counts))

leftCondition = c(rep("left", 47))

rightCondition = c(rep("right", 43))

condition = c(leftCondition, rightCondition) # Creates a "condition" vector, i.e. left or right

col\_data = data.frame(condition = condition, row.names = colnames(counts\_data)) # Creates an object with patient data as a function of left or right side

product <- function(counts\_data){

out <- 1

for(i in 1:length(counts\_data)){

out <- out\*counts\_data[i]

}

out

}

product(1:10)

rightCondition = c(rep("product", 47))

leftCondition = c(rep("product", 43))

condition = c(leftCondition, rightCondition)

col\_data = data.frame(condition = condition, row.names = colnames(counts\_data))

dds <- DESeqDataSetFromMatrix(countData = counts\_data, colData = col\_data, design = ~ 1)

dim(dds)

head(assay(dds), 1000) #can increase the value of the integer

colSums(assay(dds))

rowRanges(dds)

str(metadata(rowRanges(dds)))

colData(dds)

colData(dds) #Check to make sure it reads the conditions properly

colData(dds) <- DataFrame(col\_data) #Generates a table with conditions

colData(dds)

dds$cell #all of these should say NULL which is fine

dds$row

dds$column

dds$dex

countdata <- assay(dds)

#head(countdata, 3)

head(countdata, 1000)

#write.csv(countdata,file="countdatatargets.csv", sep ="\t", quote = FALSE)

coldata <- colData(dds)

lambda <- 10^seq(from = -1, to = 2, length = 1000)

cts <- matrix(rpois(1000\*100, lambda), ncol = 100)

meanSdPlot(cts, ranks = FALSE) #generate a graph

log.cts.one <- log2(cts + 1)

meanSdPlot(log.cts.one, ranks = FALSE) #Generate a graph

rld <- rlog(dds, blind = FALSE) #This step takes a while

head(assay(rld), 3)

vsd <- vst(dds, blind = FALSE)

head(assay(vsd), 3)

dds <- estimateSizeFactors(dds)

df <- bind\_rows(

as\_data\_frame(log2(counts(dds, normalized=TRUE)[, 1:2]+1)) %>%

mutate(transformation = "log2(x + 1)"),

as\_data\_frame(assay(rld)[, 1:2]) %>% mutate(transformation = "rlog"),

as\_data\_frame(assay(vsd)[, 1:2]) %>% mutate(transformation = "vst"))

colnames(df)[1:2] <- c("x", "y")

ggplot(df, aes(x = x, y = y)) + geom\_hex(bins = 80) +

coord\_fixed() + facet\_grid( . ~ transformation) #generates 3 different graphs

sampleDist <- dist(t(assay(rld)))

sampleDist

sampleDistMatrix <- as.matrix( sampleDist )

rownames(sampleDistMatrix) <- paste( rld$dex, rld$cell, sep = " - " )

colnames(sampleDistMatrix) <- NULL

colors <- colorRampPalette( rev(brewer.pal(9, "Blues")) )(255)

pheatmap(sampleDistMatrix,

clustering\_distance\_rows = sampleDist,

clustering\_distance\_cols = sampleDist,

col = colors)

colors <- colorRampPalette( rev(brewer.pal(7, "Greens")) )(255)

pheatmap(sampleDistMatrix,

clustering\_distance\_rows = sampleDist,

clustering\_distance\_cols = sampleDist,

col = colors)

poisd <- PoissonDistance(t(counts(dds)))

samplePoisDistMatrix <- as.matrix( poisd$dd )

rownames(samplePoisDistMatrix) <- paste( rld$dex, rld$cell, sep=" - " )

colnames(samplePoisDistMatrix) <- NULL

pheatmap(samplePoisDistMatrix,

clustering\_distance\_rows = poisd$dd,

clustering\_distance\_cols = poisd$dd,

col = colors)

dds <- DESeq(dds)

res <- results(dds)

res

summary(res)

res.05 <- results(dds, alpha = 0.05)

table(res.05$padj < 0.05)

write.csv(res.05,file=”p-valuesafterclustering.csv", sep ="\t", quote = FALSE)

dds <- DESeq(dds)

res <- results(dds)

res

summary(res)

#res.05 <- results(dds, alpha = 0.05)

#table(res.05$padj < 0.05)

#write.csv(res.05,file=”p-valuesafterclustering.csv", sep ="\t", quote = FALSE)

sum(res$pvalue < 0.05, na.rm=TRUE)

sum(!is.na(res$pvalue))

resSig <- subset(res, padj < 0.05)

head(resSig[ order(resSig$log2FoldChange), ])

head(resSig[ order(resSig$log2FoldChange, decreasing = TRUE), ])

write.csv(resSig,file="p-valuesdata.csv", sep ="\t", quote = FALSE)

counts=matrix(rpois(5400,10), ncol=6)

designs=c(0,0,0,1,1,1)

seqData=newSeqCountSet(counts, designs)

seqData

seqData=estNormFactors(seqData)

seqData=estDispersion(seqData)

result=waldTest(seqData, 0, 1)

head(result,5)

counts = matrix(rpois(600, 10), ncol=6)

designs = c(0,0,0,1,1,1)

result = DSS.DE(counts, designs)

head(result)

#write.csv(result,file="DSSpvalues.csv", sep ="\t", quote = FALSE)

padj\_r <-p.adjust(result$table$PValue)

DSS\_topGenes = which(result $padj < 0.05)

write.csv(result,file="topDSSGenes.csv", sep ="\t", quote = FALSE)

temp<-DGEList(counts=counts\_data, group=condition)

temp<-calcNormFactors(temp)

design<-model.matrix(~designs)

temp<-estimateCommonDisp(temp, design)

temp<-estimateTagwiseDisp(temp)