Lab Book 4/12/15

Claire Green

Monday

On monday I began learning how to create a list of differentially expressed genes from RNA expression data. The method I used was given to me by WenBin and I adapted it for my data.

First, you import your data and normalise it using the rma function.

```
setwd("/Users/clairegreen/Documents/PhD/TDP-43/TDP-43 Data Sets/CHMP2B/")
library(affy)
library(Biobase)
library(tkWidgets)

#run program to choose .CEL files from directory
celfiles <- fileBrowser(textToShow = "Choose CEL files", testFun = hasSuffix("[cC][eE][lL]"))
#celfiles<-basename(celfiles)
Data<-ReadAffy(filenames=celfiles) #read in files
rmaEset<-rma(Data) #normalise using RMA
analysis.name<-"C9orf72 vs C9orf72_Control" #Label analysis
dataMatrixAll<-exprs(rmaEset) #takes expression from normalised expression set
```

Next, you use the mas5calls function to indicate if the signal of a probe is (P)resent, (M)arginal or (A)bsent. This function uses the Wilcoxon signed rank test to assign a probe a p value and then threshold that p value into the three above groups.

```
#mas5call generates presence/absence calls for each probeset
mas5call<-mas5calls(Data)
callMatrixAll<-exprs(mas5call)
colnames(callMatrixAll)<-sub(".CEL", ".mas5-Detection", colnames(callMatrixAll),fixed=TRUE)
colnames(callMatrixAll)<-sub(".cel", ".mas5-Detection", colnames(callMatrixAll),fixed=TRUE)
callMatrixAll<-as.data.frame(callMatrixAll)
callMatrixAll$ProbeSetID<-rownames(callMatrixAll)
countPf<-function(x){
  sum(x=="P")
}
#count how many samples have presence calls
countPl<-apply(callMatrixAll, 1, countPf)
callMatrixAll$ProbeSetID<-rownames(callMatrixAll)
countPdf<-data.frame(ProbeSetID=names(countPl), countP=countPl)
```

The next step is to annotate the file. There are two ways presented here: Wenbin recommended using the affymetrix annotation file, which has a much better overall annotation rate (more probesets annotated) however it produces multiple genes per probeset which is often incompatible with software that interprets gene lists. The alternative that I have used is Biomart. Biomart is known to be less reliable in terms of availability of the database, and produces a lower rate of annotation, however Biomart has more customisation of attributes, and does produce 1 gene per probeset which makes it easier to use further down the line.

```
##USING BIOMART
library (biomaRt)
```

```
mart <- useMart("ENSEMBL_MART_ENSEMBL",dataset="hsapiens_gene_ensembl", host="www.ensembl.org")
x <- rownames(dataMatrixAll) #create vector containing probe IDs
#mart_attribute <- listAttributes(mart)
annotation <- getBM(attributes=c("affy_hg_u133a_2", "hgnc_symbol", "description"),
                    filters = "affy_hg_u133a_2", values = x, mart = mart)
annotation<-subset(annotation, subset=(hgnc_symbol !="")) #if no gene symbol, discount

##USING ANNOTATION FILE
annotation.file<-"/Users/clairegreen/Documents/PhD/TDP-43/TDP-43 Data Sets/HG-U133_Plus_2.na35.annot.csv
annotation<-read.table(annotation.file, header = TRUE, row.names=NULL, sep="\t", skip=0, stringsAsFactor
dim(annotation)
nrow(annotation)
#[1] 39699
annotation<-subset( annotation, subset=(Gene.Symbol !="---")) #if no gene symbol, discount
nrow(annotation)
```

Next is the main calculation of differential expression. First, a linear model is fittef for each gene in the array (lmFit). This is then used as the input for the function eBayes which computes moderated t-statistics and F-statistic, and log odds of differential expression.

```
Treat<-factor(rep(c("Control", "Patient"),c(3,8)), levels=c("Control", "Patient"))
design<-model.matrix(~Treat)
rownames(design)<-colnames(expressionMatrix)
design

#Conduct statistical analysis of expression
library(limma)
fit<-lmFit(expressionMatrix, design) #linear model fit
fit<-eBayes(fit)
result<-topTable(fit, coef="TreatPatient", adjust="BH", number=nrow(expressionMatrix)) #"BH" adjust for
#toptable normally takes top number but this takes all
```

Thursday

On Thursday I implemented the above code again, as I realised that the annotation attribute was the wrong platform for C9orf72, CHMP2B and sALS. After changing the platform I was obtaining many more results for differentially expressed genes. The problem I've been having is that each data set produces extremely different numbers of DE genes and I have been manipulating the p values to try and get a list of 1000 DE genes.

Later on Thursday I talked to John and he explained to me that it was better to idenitfy a way of taking the top 1000 genes regardless of p value (which is an arbitrary threshold that isn't so useful for us). Consequently I added the following section of code:

```
uniqueresult <- result[!duplicated(result[,15]),]
genesort <- uniqueresult[order(uniqueresult$adj.P.Val),]
topgene <- genesort[1:1000,]
write.csv(x = topgene, file = "C9orf72_anno_1000")
topgene <- genesort[1:2000,]
write.csv(x = topgene, file = "C9orf72_anno_2000")
topgene <- genesort[1:3000,]
write.csv(x = topgene, file = "C9orf72_anno_3000")
```

```
topgene <- genesort[1:4000,]
write.csv(x = topgene, file = "C9orf72_anno_4000")
topgene <- genesort[1:5000,]
write.csv(x = topgene, file = "C9orf72_anno_5000")
```

This takes the DE result for each gene and ranks them by adjusted p value. This means that the genes are ranked by how differentially expressed they are, regardless of direction (upregulated or downregultated). I took the top 1000 - 5000 genes because a consensus across so many data sets is often very difficult. The consesus was 0 for top 1000, 2(CSRP1 & RNF13) genes for top 2000, and 4 genes for top 3000. The other two I will look at tomorrow.

Friday

The overall aim of differential expression analysis was to identify if there were any common differentially expressed genes across the 5 data sets. Hopefully, these genes will have biological relevence to the mechanisms involved in TDP-43 pathology, and even better they will correspond with the pathprint signature that I have already generated. The methodology I have been using is thus: take the top X000 genes from each data set and generate a consensus list of genes. With this list of genes, you can attempt validation in four different ways.

The first way is to validate mathematically by conducting random permutations of the equivalent gene list size (e.g. for a list generated from top 2000 genes, validate by picking 2000 random genes 10,000 times, and if significant, less consensus genes will be generated on average than the experimental list)

You can also look at the list and identify if there are semantic links: are these genes shown to be linked to disease? E.g. in GWAS studies? If not, do they play a role in mechanisms that are known to be dysregulated in the diseases of interest?

A method closely linked to that above is to identify whether any genes in the list are present in the gene lists of pathways in my pathprint signature. This would link the list biologically to the signature that I am trying to validate. This is dependent on positive results from the statistical validation above as alone, this kind of concordance is very weak.

Finally, John suggested a method by which I perform pairwise correlation analysis between every gene in the top X000 with every other gene. Those that have high correlations with many genes tend to be more functionally relevent, therefore if I can prove that genes in my list are highly correlatory, it suggests they are more than just noise that happened to be selected by Type I error