Skeleton Key for RNAseq analysis

Written By: Ciera Martinez

About

This is the script used to perform differential gene expression analysis using `edgeR`. The tissue came from p4 leaves of *Solanum lycopersicum* using Laser Capture Microdissection.

Key to Samples

genotype: either wildtype of *tf2*

region: A. tip B. early emmerging leaflet C. base

type: MBR = Marginal Blastozone Region, other = the rachis or midvein region

Samples:

```
print(sample1)
```

```
## [1] "tf2aother"
```

```
print(sample2)
```

```
## [1] "wtaother"
```

See README.md for more detailed instructions of how to use script

Run the `render()` function below and everything will be run with report at end.

```
library(rmarkdown)
render("skeletonDE.Rmd", "pdf_document", output_file = paste(sample1,"_",sample2,"_","DE.pdf",sep=""))
```

Analysis (Acutal start)

libraries

```
library(edgeR)
library(yaml)
```

Read in YAML guide

This reads in the information in the `de.yml` file which has the two names of the samples you are interested in comparing.

```
yamls <- yaml.load_file("de.yml")
```

This part assigns your YMAL to a object in R. This will be used throughout the script to specify which sample types you are comparing.

```
sample1 <- yamls$sample1
sample2 <- yamls$sample2

sample1
```

```
## [1] "tf2ambr"
```

```
sample2
```

```
## [1] "wtambr"
```

Read in Data

Read in raw count data per gene.

```
counts <- read.delim("../requisiteData/sam2countsResults.tsv",row.names=1)

#check the file
head(counts)
colnames(counts)
#need to convert NA to 0 counts
counts[is.na(counts)] <- 0
```

Subset DE expirement

Start by subsetting the particular treatments which are being compared. This might need to be modified depending on the naming of your samples. In my case each sample is named by sample and rep number, so the script is identifying any sample with the sample name given in the `de.yml` file.

```
colnames(counts)
```

```
##  [1] "tf2ambr1"      "tf2ambr3"      "tf2ambr4"      "tf2ambr6"
##  [5] "tf2aother1"    "tf2aother2"    "tf2aother4"    "tf2aother7"
##  [9] "tf2bmbr2"      "tf2bmbr5"      "tf2bmbr6"      "tf2bother1"
## [13] "tf2bother3"    "tf2bother4"    "tf2bother6"    "tf2cmbr1.4"
## [17] "tf2cmbr3"      "tf2cmbr6"      "tf2cmbr7"      "tf2cother2"
## [21] "tf2cother5"    "tf2cother6"    "tf2cother7"    "wtambr2"
## [25] "wtambr4"       "wtambr5"       "wtaother1"     "wtaother5"
## [29] "wtaother6"     "wtaother7"     "wtaother8"     "wtbmbr2"
## [33] "wtbmbr3"       "wtbmbr6"       "wtbmbr8"       "wtbother1.4"
## [37] "wtbother3"     "wtbother5"     "wtbother8"     "wtcmbr10"
## [41] "wtcmbr1.4.6"   "wtcmbr2"       "wtcmbr3"       "wtcmbr7"
## [45] "wtcmbr9"       "wtcother1.3.4" "wtcother2"     "wtcother6"
```

```
counts1 <- counts[,grep(sample1, colnames(counts), value = TRUE)]
count1Len <- length(colnames(counts1)) #used in to specify library group in next step.

counts2 <- counts[,grep(sample2, colnames(counts), value = TRUE)]
count2Len <- length(colnames(counts2)) #used to specify library group in next step.

counts <- cbind(counts1, counts2)

head(counts)
```

```
##                     tf2ambr1 tf2ambr3 tf2ambr4 tf2ambr6 wtambr2 wtambr4
## Solyc00g005040.2.1       12        0        3       12       0       2
## Solyc00g005050.2.1       33        1       14       17       0       6
## Solyc00g005060.1.1        1        5        1        1       0       0
## Solyc00g005070.1.1       14       22       23        5      24       3
## Solyc00g005080.1.1       19        2       25       32       9      15
## Solyc00g005150.1.1        3        0        0        4       0       1
##                     wtambr5
## Solyc00g005040.2.1       8
## Solyc00g005050.2.1       6
## Solyc00g005060.1.1       1
## Solyc00g005070.1.1       9
## Solyc00g005080.1.1      19
## Solyc00g005150.1.1       2
```

Add column specifying library Group

Make a vector called group that will be used to make a new column named group to identify library region type.

```
group <- c(rep(sample1, count1Len), rep(sample2, count2Len))
d <- DGEList(counts=counts,group=group)
```

Check to see if the group column matches your sample name and they are appropriate.

```
d$samples
```

```
##            group lib.size norm.factors
## tf2ambr1 tf2ambr  1313540            1
## tf2ambr3 tf2ambr    91726            1
## tf2ambr4 tf2ambr  1438416            1
## tf2ambr6 tf2ambr  1088653            1
## wtambr2   wtambr   395165            1
## wtambr4   wtambr   792542            1
## wtambr5   wtambr   632686            1
```

Differential expression using edgeR

Make sure there is full understanding of each edgeR command being used. The manual is amazing so read it *before* running the DE analysis below edgeR manual. There are many options and they must be set to be appropriate for your analysis.

```
cpm.d <- cpm(d) #counts per mutant
d <- d[rowSums(cpm.d>5)>=3,] #This might be a line to adjust. It is removing genes with low counts.
d <- estimateCommonDisp(d,verbose=T)
```

```
## Disp = 0.432 , BCV = 0.6573
```

```
d <- calcNormFactors(d)
d <- estimateCommonDisp(d)

DEtest <- exactTest(d,pair=c(sample1,sample2))
head(DEtest$table)
```

```
##                       logFC logCPM PValue
## Solyc00g005040.2.1 -0.1208  3.194 1.0000
## Solyc00g005050.2.1 -1.1173  3.896 0.1451
## Solyc00g005070.1.1 -0.1168  5.566 0.8295
## Solyc00g005080.1.1  0.7194  4.803 0.3426
## Solyc00g005160.1.1 -0.1018  2.784 0.6612
## Solyc00g005440.1.1 -0.9405  4.828 0.2010
```

```
results <- topTags(DEtest, n=Inf)
head(results)
```

```
## Comparison of groups:  wtambr-tf2ambr
##                     logFC logCPM    PValue       FDR
## Solyc08g079850.1.1 6.596  9.407 7.328e-16 1.036e-11
## Solyc12g010020.1.1 6.344  6.947 1.890e-13 1.336e-09
## Solyc00g187050.2.1 6.572  6.619 4.085e-13 1.925e-09
## Solyc06g024350.1.1 5.578  8.390 1.721e-12 5.314e-09
## Solyc03g062850.1.1 6.467  7.246 1.879e-12 5.314e-09
## Solyc03g098790.1.1 6.287  6.238 3.996e-12 9.416e-09
```

```
dim(results$table)
```

```
## [1] 14139     4
```

```
sum(results$table$FDR<.05) # How many are DE genes?
```

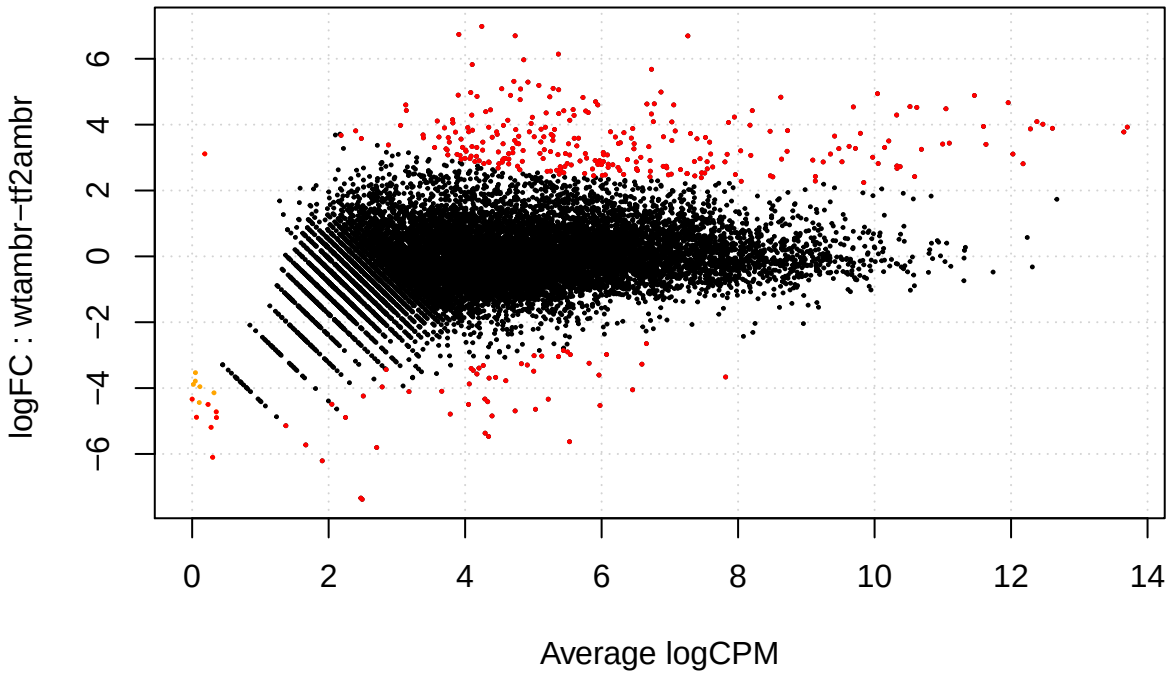
```
## [1] 330
```

```
summary(decideTestsDGE(DEtest,p.value=.05))
```

```
##    [,1]
## -1   57
## 0  13809
## 1    273
```

```
sig.genes <- rownames(results$table[results$table$FDR<0.05,]) # outputs just significant gene names

plotSmear(d,de.tags=sig.genes)
```



Subset all the genes with a significant FDR score less than .05.

```
results.sig <- subset(results$table, results$table$FDR < 0.05)
```

dim(results.sig)

What are the genes that are misexpressed? For this we need to add some annotation.

Essentially we are merging two annotations files to 1.) only sig genes 2.) all genes

```
annotation1<- read.delim("../requisiteData/ITAG2.3_all_Arabidopsis_ITAG_annotations.tsv", header=FALSE)
colnames(annotation1) <- c("ITAG", "SGN_annotation")
annotation2<- read.delim ("../requisiteData/ITAG2.3_all_Arabidopsis_annotated.tsv")
annotation <- merge(annotation1,annotation2, by = "ITAG")
head(annotation)
```

```
##                 ITAG
## 1 Solyc00g005000.2.1
## 2 Solyc00g005040.2.1
## 3 Solyc00g005050.2.1
## 4 Solyc00g005080.1.1
## 5 Solyc00g005900.1.1
## 6 Solyc00g006490.2.1
##
## 1                                                    Aspartic proteinase nepenthesin I (AHRD V1 **--
## 2                                                       Potassium channel (AHRD V1 ***- D0EM91_9ROSI)
## 3
## 4
```

```
## 5 Oxygen-evolving enhancer protein 1, chloroplastic (AHRD V1 ***- PSBO_SOLTU); contains Interpro doma
## 6    Serine/threonine-protein phosphatase 6 regulatory subunit 3 (AHRD V1 **-- SAPS3_HUMAN); contains
##         AGI symbol
## 1 AT3G20015   <NA>
## 2 AT5G46240   KAT1
## 3 AT5G11680   <NA>
## 4 ATCG01280 YCF2.2
## 5 AT5G66570  MSP-1
## 6 AT1G07990   <NA>
##
## 1
## 2
## 3
## 4
## 5
## 6 SIT4 phosphatase-associated family protein; similar to SIT4 phosphatase-associated family protein
##   X..identity alignment.length e.value bit.score percent.query.align
## 1       63.76              447  7e-148       520               89.94
## 2       66.02              103   2e-37       150               85.71
## 3       76.96              204   1e-88       322               98.98
## 4       91.25               80   2e-38       153               79.80
## 5       69.62               79   4e-26       112               78.79
## 6       61.92              856   0e+00       979               99.77
```

```
#Making the only significant gene table
results.sig$ITAG <- rownames(results.sig)  #change row.names to ITAG for merging
results.sig.annotated <- merge(results.sig,annotation,by = "ITAG", all.x=TRUE) #This is merging only si

#Making all table

results$table$ITAG <- rownames(results$table)
results.all.annotated <- merge(results$table, annotation,by = "ITAG")
```

Write table with results.

```
write.table(results.all.annotated, file=paste(sample1,"_",sample2,"_","DE_all.txt",sep=""),sep="\t",row
write.table(results.sig.annotated, file=paste(sample1,"_",sample2,"_","DE_sig.txt",sep=""),sep="\t",row
```

Now run the script below for a full `knitr` report of what was run and leave this report in the folder that the analysis was done with output files.